User's Manual

32

FDP Manager for Linux

User's Manual: Software

R-Car H2/M2 Series

All information contained in these materials, including software and software specifications, represents information on the software at the time of publication and is subject to change by Renesas Electronics Corp. without notice. Please review the latest information published by Renesas Electronics Corp. through various means, including the Renesas Electronics Corp. website (http://www.renesas.com).

Renesas Electronics
www.renesas.com

Rev.1.01 Jun. 2014

Notice

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

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

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

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

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

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

(2012.4)

How to Use This Manual

1. Purpose and Target Readers

This manual is designed to provide the user with an understanding the functions of this driver to management FDP H/W resource and for the reference manual to develop systems implementing IP conversion function. This manual is written for engineers who use this FDP management functions with FDP.

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

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

Please refer to documents of softwares and hardware for a target system implementing this FDP Manager driver as necessary.

The following documents are related documents. Make sure to refer to the latest versions of these documents. The newest versions of the documents listed may be obtained from the Renesas Electronics Web site.

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

2. Notation of Numbers and Symbols

This manual use following notation.

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

3. List of Abbreviations and Acronyms

Abbreviation	Full Form
FDP	Fine Display Processor
IPC	Interlace Progressive Conversion
2D-IPC	2 dimension Interlace Progressive Conversion
3D-IPC	3 dimension Interlace Progressive Conversion
RGB	R(red)-G(green)-B(blue) color format
YUV	YUV color format
API	Application Programing Interface
POSIX	Portable Operating System Interface
DTV	Digital Television
DVD	Digital Versatile Disc

Table of Contents

1. Overview

1.1 Overview of this Software

This document is described about FDP Manager. FDP Manager is driver of FDP(Fine Display Processor, interlace progressive conversion module) hardware resource.

FDP manager is the mechanism of IP conversion H/W(FDP) resource management for high efficiency use. FDP manager categorize application use condition (frame sync/async and H/W occupy/common use) and assign H/W according to categorized condition.

1.2 Configuration of this Software

This software consists of the following resources.

- Documents
- Source code
- Make file

Table 1.1 and Figure 1-1 show the configurations of the released software.

To use this software, the following additional software which is not included in this software is required.

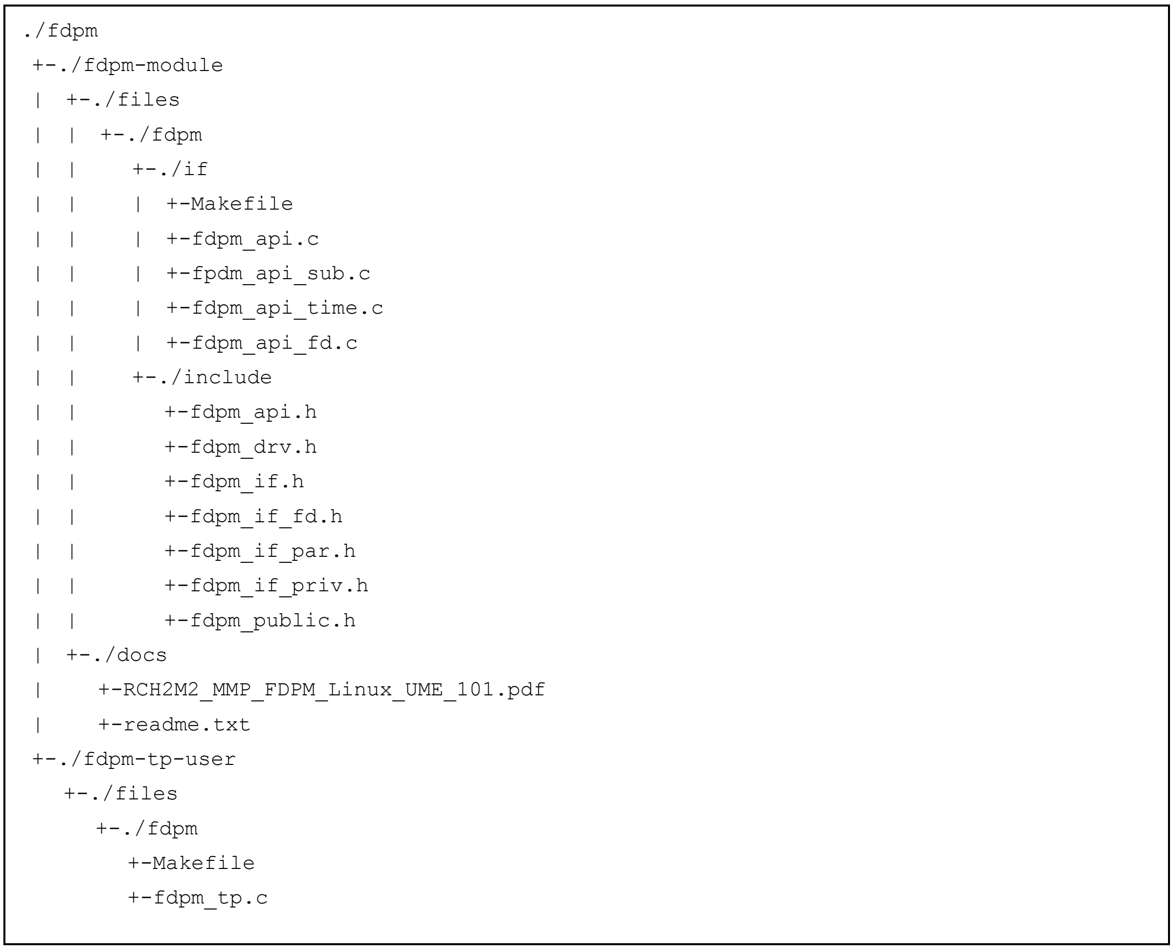
Details of this additional software are shown below.

- Kernel module source code

This software is distributed in based on Dual MIT/GPLv2 licenses. Figure 1-2 shows the list of these source files.

Table 1.1 configuration of document file

No.	Name
1	R-Car H2/M2 Series FDP Manager for Linux User’s Manual (this document)



```
./fdpm
 +-./fdpm-module
 |  +-./files
 |  |  +-./fdpm
 |  |     +-./if
 |  |     |  +-Makefile
 |  |     |  +-fdpm_api.c
 |  |     |  +-fpdm_api_sub.c
 |  |     |  +-fdpm_api_time.c
 |  |     |  +-fdpm_api_fd.c
 |  |     +-./include
 |  |        +-fdpm_api.h
 |  |        +-fdpm_drv.h
 |  |        +-fdpm_if.h
 |  |        +-fdpm_if_fd.h
 |  |        +-fdpm_if_par.h
 |  |        +-fdpm_if_priv.h
 |  |        +-fdpm_public.h
 |  +-./docs
 |     +-RCH2M2_MMP_FDPM_Linux_UME_101.pdf
 |     +-readme.txt
 +-./fdpm-tp-user
    +-./files
       +-./fdpm
          +-Makefile
          +-fdpm_tp.c
```

Figure 1-1 configuration of this software

```
./fdpm
 +-./fdpm-module
   +-./files
      +-./fdpm
        +-./drv
        |  +-./fdp
        |  |  +-fdp_depend.h
        |  |  +-fdp_drv.c
        |  |  +-fdp_drv.h
        |  |  +-fdp_drv_hardw.h
        |  |  +-fdp_drv_imgset.c
        |  |  +-fdp_drv_l.c
        |  |  +-fdp_drv_l.h
        |  |  +-fdp_drv_lfunc.h
        |  |  +-GPL_COPYING
        |  |  +-MIT_COPYING
        |  +-./include
        |  |  +-fdpm_def.h
        |  |  +-fdpm_depend.h
        |  |  +-fdpm_lfunc.h
        |  |  +-fdpm_log.h
        |  |  +-fdpm_main.h
        |  |  +-GPL_COPYING
        |  |  +-MIT_COPYING
        |  +-./manager
        |  |  +-fdpm_resource.c
        |  |  +-fdpm_seq.c
        |  +-fdpm_api.h
        |  +-fdpm_ctrl.c
        |  +-fdpm_if.h
        |  +-fdpm_if_par.h
        |  +-fdpm_ioctl.c
        |  +-fdpm_main.c
        |  +-fdpm_public.h
        |  +-fdpm_sub.c
        |  +-fdpm_task.c
        |  +-GPL_COPYING
        |  +-Makefile
        |  +-MIT_COPYING
        +-./include
           +-GPL_COPYING
           +-MIT_COPYING
           +-fdpm_drv.h
           +-fdpm_api.h
           +-fdpm_public.h
```

Figure 1-2 configuration of kernel source code

1.2.1 FDP Manager Structure

FDP Manager function structure shows in Figure 1-3.

This software and kernel layer driver cooperate to function FDP management and IP conversion.

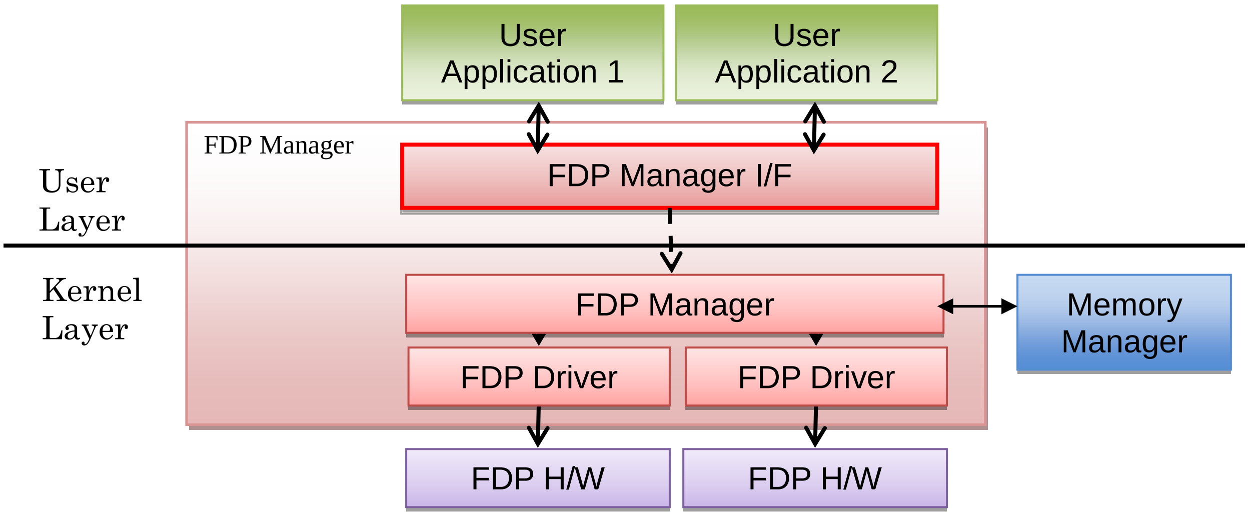


Figure 1-3 FDP Manager structure

1.2.2 Specifications

Table 1.2 lists the Specifications.

Table 1.2 Specifications

Item	Function	Specification
Management function	Resource management	FIFO type scheduling
	Timing mode	V-sync mode/Best Effort mode
Driver function	Input format	NV12,NV21,NV16,YV12,YU12,YUY2
	Output format	NV12,NV21,NV16,YV12,YU12,YUY2,UYVY

	Input size	[interlace]
		Min:80pixel(H)x 40pixel(V) Max:1920pixel(H)x 960pixel(V)
		[progressive]
		Min:80pixel(H)x 80pixel(V) Max:1920pixel(H)x1920pixel(V)

	Sequence mode	Progressive/Interlace full-rate/interlace half-rate
	Conversion mode	3D-IPC/2D-IPC
	Pull-down	2:3 pull-down/2:2 pull-down
	Field-delay	0V(progressive/2D-IPC)/1V(3D-IPC)

1.2.3 Condition

(1) Memory Manager

FDP Manager use memory manager driver kernel API for allocate memory of FDP H/W internal use. In case of using FDP Manager, load memory manager driver in advance.

(2) Linux kernel configuration

FDP Manger use POSIX timer for generating V-sync timing. To achieve 0.1ms order resolution V-sync, it is necessary to enable High Resolution Timer support in Linux Kernel configuration.

(3) Memory allocation

Memory allocation of FDP manager specify below.

(a) kmalloc use for variables in kernel layer.

(b) Memory manager kernel API for FDP H/W. FDP H/W accesses to memory as temporary buffer for 3D-IPC. FDP Manager allocates this buffer by using memory manager kernel API at drv_FDP_Open call and releases this at drv_FDP_Close call. Allocation size for one stream is below.

Allocation_size (byte) = 2 x ((input_width(pixel) + 7)>>3) x input_height(pixel)

For example, in case of 1920x1080 output size, input size is 1920x540.

Allocation size is 2 x ((1920+7)>>3) x 540 = 259200(byte)

(4) H/W resource

FDP Manager use two FDPs for R-CarM2, three FDPs for R-CarH2.

(5) FDP management resource define

For FDP resource management, number of FDP module in target LSI, permit number of Applications to one FDP H/W. Define name display in table 1.3. These define parameter are configured at build by environment variable "FDPM_CONFIG" define "H2CONFIG" or "M2CONFIG".

Table 1.3 define table

Define name	value		Description	Available value(*1)
	R-CarH2	R-CarM2		
FDPM_FDP_NUM	3	2	Number of FDP modules in target LSI 2: R-CarM2 3: R-CarH2	Do not change default value.
FDPM_VINT_MODE_NUM	2	2	Permit number of assignment applications for one FDP in common use Vint mode.	Depend on required number and performance of common use Vint mode application.
FDPM_BE_HW_NUM	2	1	Permit using number of FDP H/W for best effort mode (FDP_VMODE_VBEST) applications.	From 1 to FDPM_FDP_NUM

(*1)If you change from define value, please consider required performance at the use case on your system.

1.3 Development Environment

1.3.1 Hardware development environment

The example of hardware required for the development is shown in the following table.

Table 1.4 hardware used by the development

Hardware Name		Remarks

Platform	RTP0RC7790SEB00010S(LAGER)
	RTP0RC7791SEB00011S(KOELSCH)
		-

Device	R-Car H2/M2	-
Using IP	FDP	-

1.3.2 Software development environment

The example of software required for the development is show in the following table.

Table 1.5 software used by the development

Software	Version /Revision	Remarks
R-Car H2/M2 Linux BSP	-	-
Memory manager	-	-

2.Installation Procedures

FDP Manager is provided source code (in kernel layer) and shared library (in user layer).

2.1.1 Building the shared library

The following is the procedure for building the shared library that are included in this software.

[1] Setting environment variables

Set the following environment variables. Define $WORK is work directory. CROSS_COMPILE variable setting is an example in case the cross compiler extracting directory is $WORK.

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

Note: the ‘XXX’ is the cross compiler version number. Please follow the instructions of BSP.

[2] Check Build option

FDP Manager has parameter check function. If you want disable parameter check,

remove -DFDP_PAR_CHECK_MODE option in Makefile under “$WORK/fdpm/fdpm-module/files/fdpm/if”.

[3] Execute “make” in the build directory

```
$ cd $WORK/fdpm/fdpm-module/files/fdpm/if
$ make
```

[4] Verifying the binary module

Make sure that the following binary modules are built under “$WORK/fdpm/fdpm-module/files/fdpm/if”.

libfdpm.so.1.0.0
libfdpm.so.1 (symbolic link)
libfdpm.so (symbolic link)
Note) The symbolic link files referred when build application.

2.1.2 Building the kernel and the kernel modules

[1] Setting environment variables

Same as building the kernel modules, refer to section 2.1.1. And set the following environment variables.

Memory manager build directory (Module.symvers file exists directory. Example, $WORK/mmngr/kernel/drv) set to FDPM_MMNGRDIR variables.

```
$ export FDPM_MMNGRDIR=$WORK/mmngr/kernel/drv
```

In case of R-CarH2

```
$ export FDPM_CONFIG=H2CONFIG
```

In case of R-CarM2

```
$ export FDPM_CONFIG=M2CONFIG
```

[2] Building the Kernel

FDP manager use high resolution timer for V-sync mode application. If you want to use V-sync mode application, follow the following steps.

[2-1] Modify the kernel configuration with “make menuconfig”.

Set enable to “Kernel Features->[*]High Resolution Timer Support”.

[2-2]Build the kernel according to 5.Kernel Build in “Linux BSP Start-Up Guide”.

[3] extract Memory manager kernel module source code to work directory($WORK).

[4] extract FDP manager kernel module source code to work directory($WORK).

[5] execute “make” in the build directory

```
$ cd $WORK/fdpm/fdpm-module/files/fdpm/drv
$ make
```

[6] verifying the kernel module

Make sure that the following kernel modules are built under “fdpm/fdpm-module/files/fdpm/drv”.

fdpm.ko

2.1.3 Binary inclusion procedure

The following is the procedure for including the kernel and binary modules that are built according to the procedure described in section 2.1.1 and 2.1.2.

[1] Storing the kernel modules

Copy ‘fdpm.ko’ to BSP user land. Define $NFS is root directory on BSP.

```
$sudo cp fdpm.ko $NFS/home/root/workspace
```

[2] Storing the binary module

Copy ‘libfdpm.so.1.0.0’ to BSP user land.

Execute on PC side.

```
$ sudo cp libfdpm.so.1.0.0 $NFS/usr/local/lib
```

Execute on lager/koelsch board side.

```
$ ln -s /usr/local/lib/libfdpm.so.1.0.0 libfdpm.so.1
$ ln -s /usr/local/lib/libfdpm.so.1 libfpdm.so
```

[3] Setting environment variable on lager board side.

Set the LD_LIBRARY_PATH environment variable if ‘/usr/local/lib’ is not included in this variable.

```
$ export LD_LIBRARY_PATH=/usr/local/lib
```

2.1.4 Build user application procedure

In case of compile the user application witch link to this library, set compiler option which the directory path to the include header files directory with –I option, the shared library with –L option, and set the library name with –l option(-lfdpm).

2.1.5 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 sample program

Execute "make" in the build directory

```
$cd fdpm/fdpm-tp-user/files/fdpm
```

```
$make
```

[3] Verifying the executing object

Make sure that the following executing object is built under "fdpm/fdpm-tp-user/files/fdpm"

fdpm_tp

[4] Executing on evaluation board

Copy 'fdpm_tp' to BSP user land. And execute.

```
$./fdpm_tp
```

3. Processing Specifications

FDP Manager is the mechanism of FDP H/W resource management for high efficiency use. For this purpose, FDP Manager is got application use condition (frame sync/async and H/W resource occupy/common use) at Open operation. FDP Manager assigns H/W according to this information.

In this chapter, describe about mode definition, resource management and how to use FDP manager by using example.

3.1 Mode definition

FDP Manager manages H/W resource by using two information, V-sync mode and occupy mode. In this section, describe about mode definition.

3.1.1 V-sync mode

V-sync mode has two mode, Vint mode and Best Effort mode.

(1) Vint mode

Vint mode timing chart shows Figure 3-1.

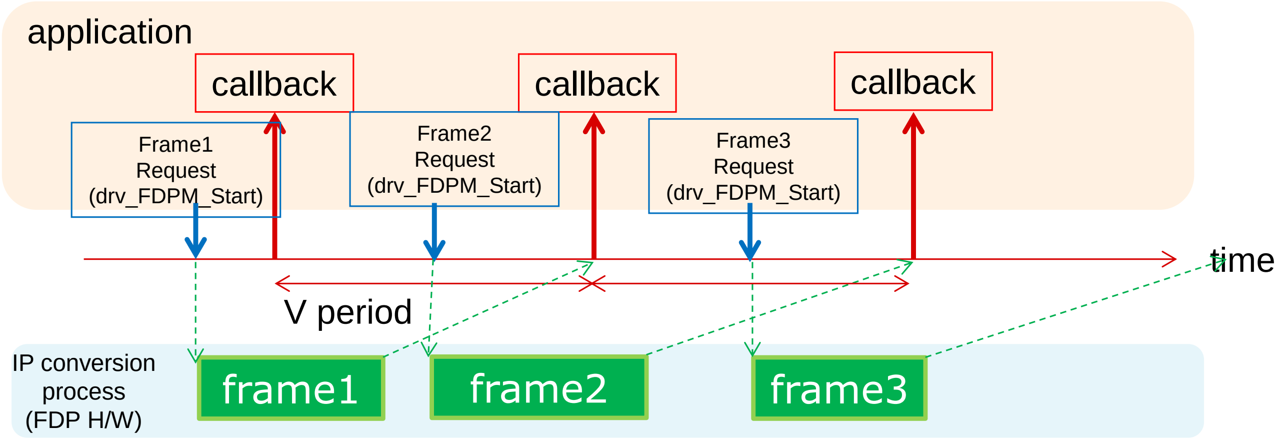

application
callback
callback
callback
Frame1 Request (drv_FDPM_Start)
Frame2 Request (drv_FDPM_Start)
Frame3 Request (drv_FDPM_Start)
time
V period
IP conversion process (FDP H/W)
frame1
frame2
frame3

Figure 3-1 Vint mode timing chart

FDP Manager returns callback at the timing that specified period (V-period) from application. Application achieve IP conversion with synchronized frame timing by requesting next frame process after receive each callback. This mode is suitable for DTV/DVD application. If you use Vint mode for your application, set FDP_VMODE_NORMAL to vmode parameter in T_FDP_OPEN struct.

(2) Best Effort mode

Best Effort mode timing char shows Figure 3-2.

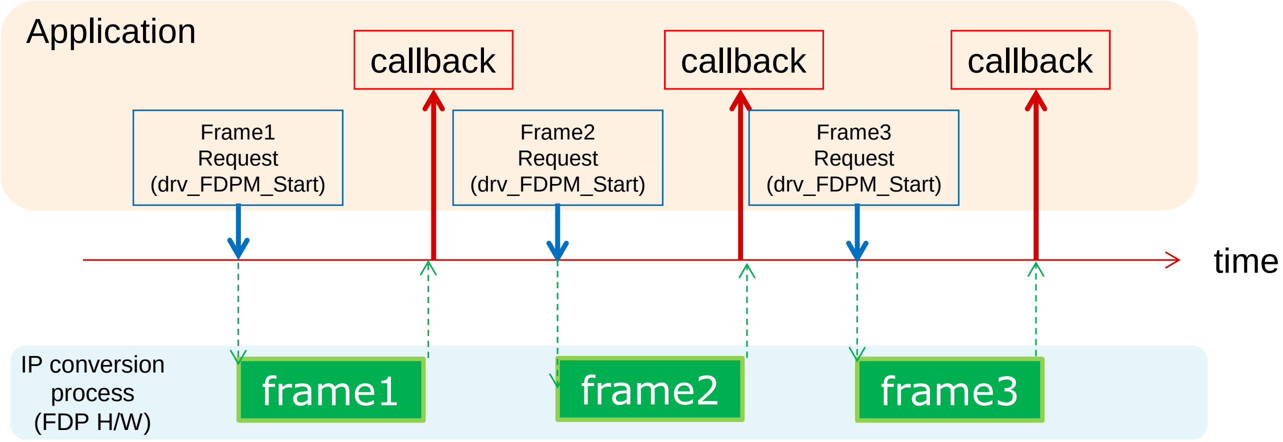

Application
callback
callback
callback
Frame1 Request (drv_FDPM_Start)
Frame2 Request (drv_FDPM_Start)
Frame3 Request (drv_FDPM_Start)
time
IP conversion process (FDP H/W)
frame1
frame2
frame3

Figure 3-2 Best Effort mode timing chart

FDP Manager return callback at the timing when finish IP conversion process by FDP H/W. This mode is suitable for unnecessary frame synchronize application. If you use Best Effort mode for your application, set FDP_VMODE_VBEST to vmode parameter in T_FDP_OPEN struct.

3.1.2 occupy mode

Occupy mode have two mode, OCCUPY mode and COMMON use mode.

(1) OCCUPY mode

OCCUPY mode shows Figure 3-3.

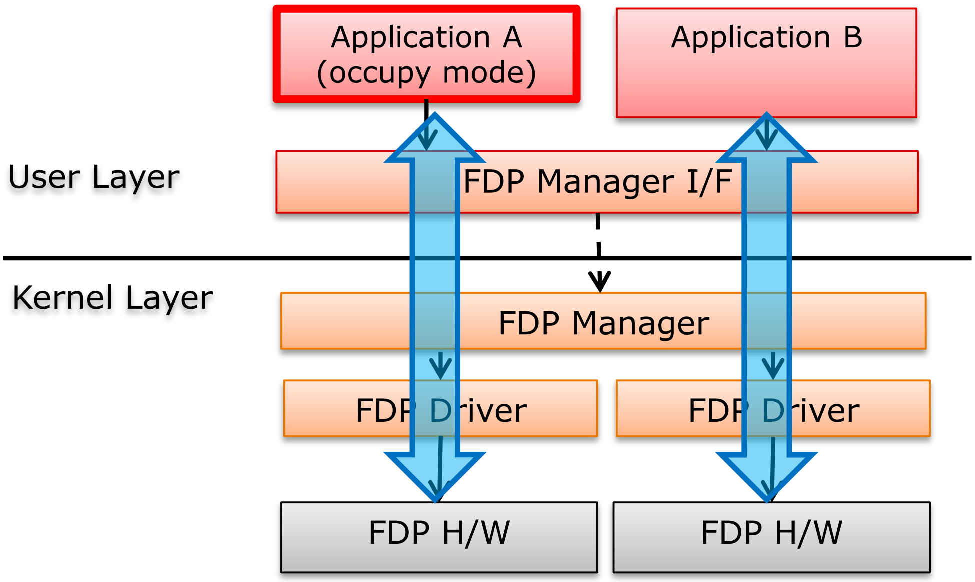


Figure 3-3 OCCUPY mode

In this mode, an application occupies each FDP H/W. This mode is suitable for high performance use case(ex. full-HD IPC, frame async mode application etc.).

(2) Common use mode

Common use mode shows Figure 3-4.

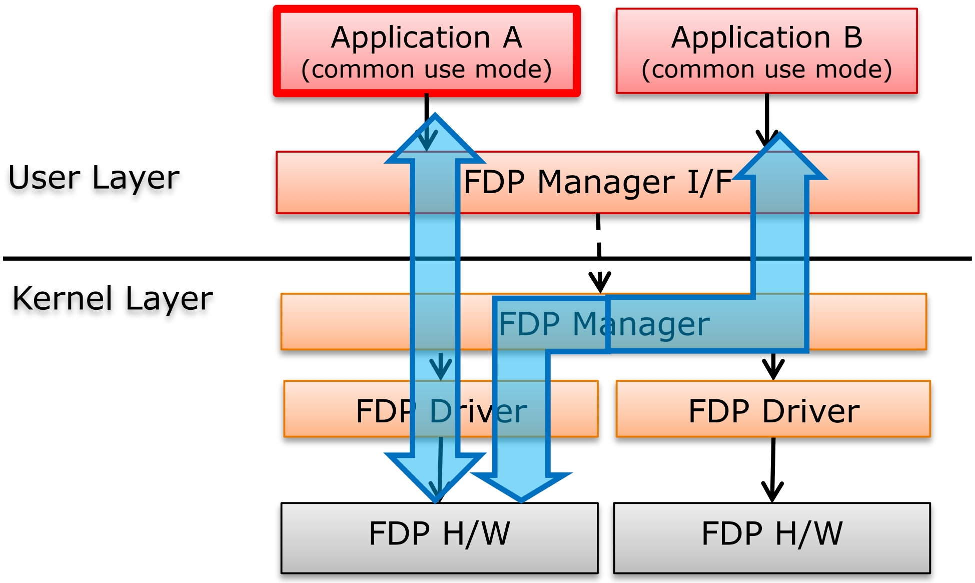


Figure 3-4 common use mode

Two common use mode applications use one FDP H/W. In case of low load use case (ex. SD size IPC), use this mode for effective FDP H/W resource. Application only can use this mode with Vint mode.

3.2 Resource management

FDP Manager support following combination of mode.

table 3.1 support combination of mode

	Vint mode	Best Effort mode
OCCUPY	Permit	Permit
Common use	Permit	N/A

FDP Manager does not support combination of common use mode and best effort mode.

(1) Combination of common use mode application

In common use mode, only Vint mode available.

Because Vint mode application affect timing by best effort mode application, best effort mode application can not available common use mode.

table 3.2 combination of common use mode application

		Application B	
		Vint mode	Best effort mode
Application A	Vint mode	Permit	N/A
	Best effort mode	N/A	N/A

Under this condition, next cause shows the max use case example in case of R-Car M2. you can adapt the same way to assumption use case of R-Car H2.

(2) Assumption use case of R-Car M2

R-Car M2 have two FDP H/W, max use case example shows bellow in each mode. FDP Manager assigns automatically which FDP H/W use.

table 3.3 assumption use case of R-Car M2

contents	mode		Use case (example)					
	Occupy/ Common	Vint/ Best effort	Ex.1	Ex.2	Ex.3	Ex.4	Ex.5	Ex.6
Apli A	Occupy	Vint	🟨 Use	🟨 Use		🟨 Use		
Apli B	Occupy	Best effort	🟧 Use		🟨 Use		🟨 Use	
Apli C	Occupy	Vint		🟧 Use				
Apli D	Occupy	Best effort			🟨 Use			
Apli E	Common	Vint				🟧 Use	🟧 Use	🟨 Use
Apli F	Common	Vint				🟧 Use	🟧 Use	🟨 Use
Apli G	Common	Vint						🟧 Use
Apli H	Common	Vint						🟧 Use

🟨 FDP0 🟧 FDP1

FDP Manager returns error when more other application uses FDP in above conditions.

3.3 How to use FDP Manager

3.3.1 Basic procedure

At design user application, user decide use mode (occupy/common use mode and Vint/Best Effort mode, describe at chapter.3) by application character. And fix use mode in the application.

User application is archive IP conversion by using FDP Manager I/F API (describe more detail at chapter 4). Basic processing procedure describe here.

At first, user application calls drv_FDPM_Open function for reserve FDP H/W resource. At call this function, open parameter (structure T_FDP_OPEN) set for specifying reserve resource. If user application select occupy mode and best effort mode, user set sub_ercd that is one of drv_FDPM_Open argument for holding file descriptor by each stream. If successes reserve FDP H/W resource, return 0. If user application select occupy mode and best effort mode, callback2 and callback3 argument should set NULL. If user application select Vint mode, callback2 should be set.

Next, user application calls first drv_FDPM_Start function for start IP conversion. If user application select Vint mode, after call drv_FDPM_Open, first callback (callback2) returned, user application calls first drv_FDP_Start function. At first time call, start parameter (structure T_FDP_START) should specify each parameter. Especially, sequence parameter (start_par->fproc_par->seq_par) should set each member. FDP Manager recognize new sequence start by seq_par specify, and recognize continue same sequence by not specify (NULL).

When IP conversion finished (best effort mode) or V-sync period passed (Vint mode), FDP Manager return callback (callback2 or callback3). When user application receives this callback, call next drv_FDP_Start function for next frame processing. IP conversion is achieved repeat this drv_FDPM_Start-callback procedure.

At the end of IP conversion, after return callback previous drv_FDPM_Start, user application calls drv_FDPM_Close function. FDP Manager release FDP H/W resource.

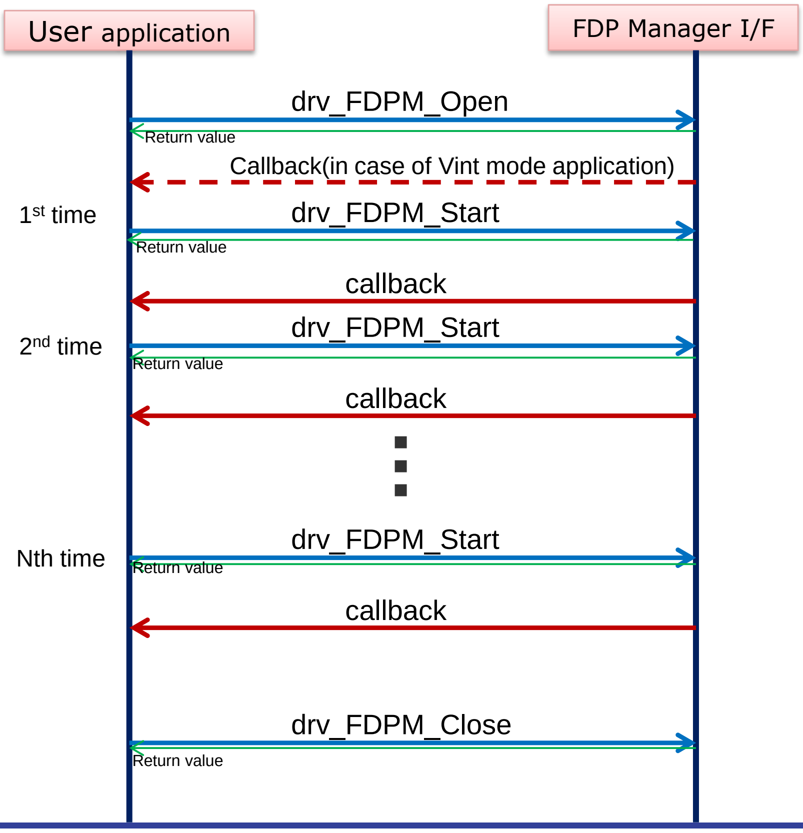


Figure 3-5 Basic processing procedure

3.3.2 Status value

User application can get FDP Manager Status information by using drv_FDPM_Status function. Status information (structure T_FDP_STATUS) described in chapter.4.

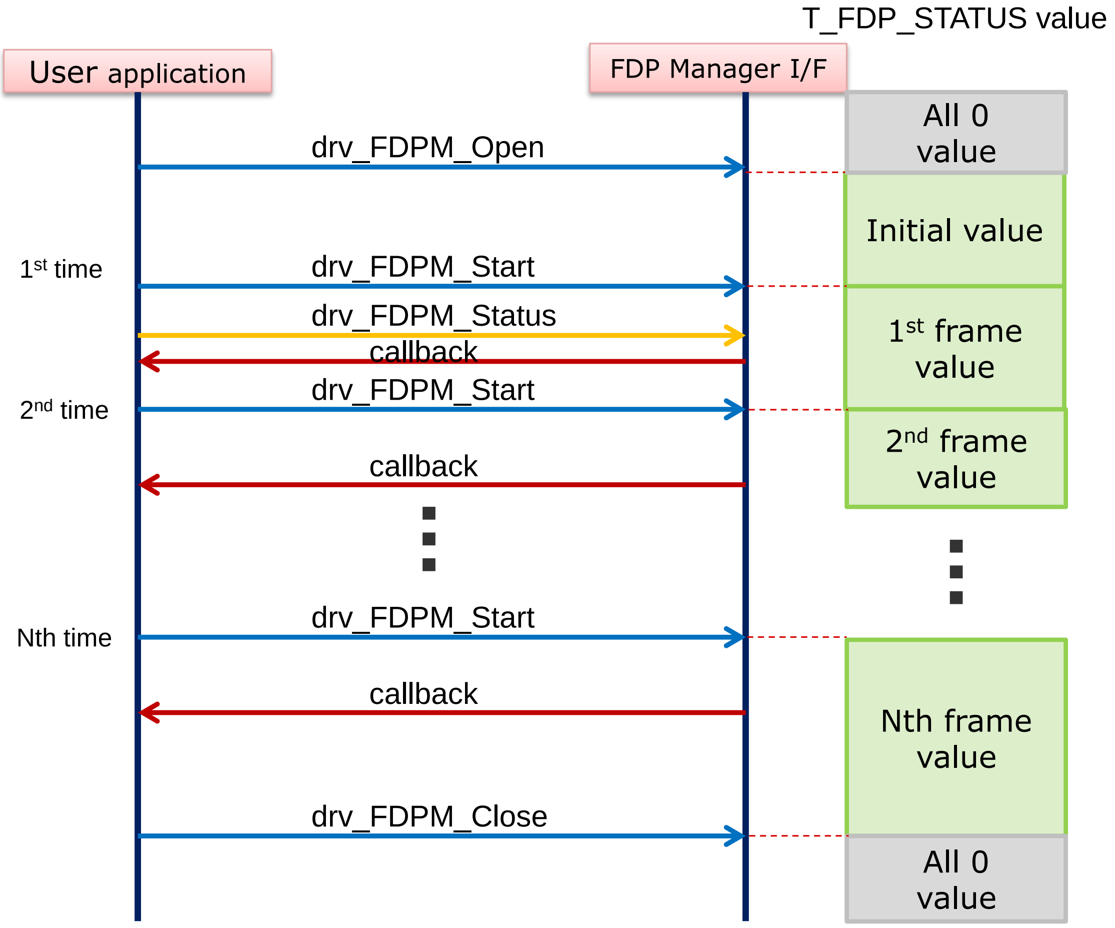


Figure 3-6 T_FDP_STATUS value transition

Status information is changed by use drv_FDPM_Status timing. If user application uses drv_FDPM_Status before drv_FDPM_Open or after drv_FDPM_Close, Status information is all 0. Status information is updated by call timing of drv_FDPM_Start function.

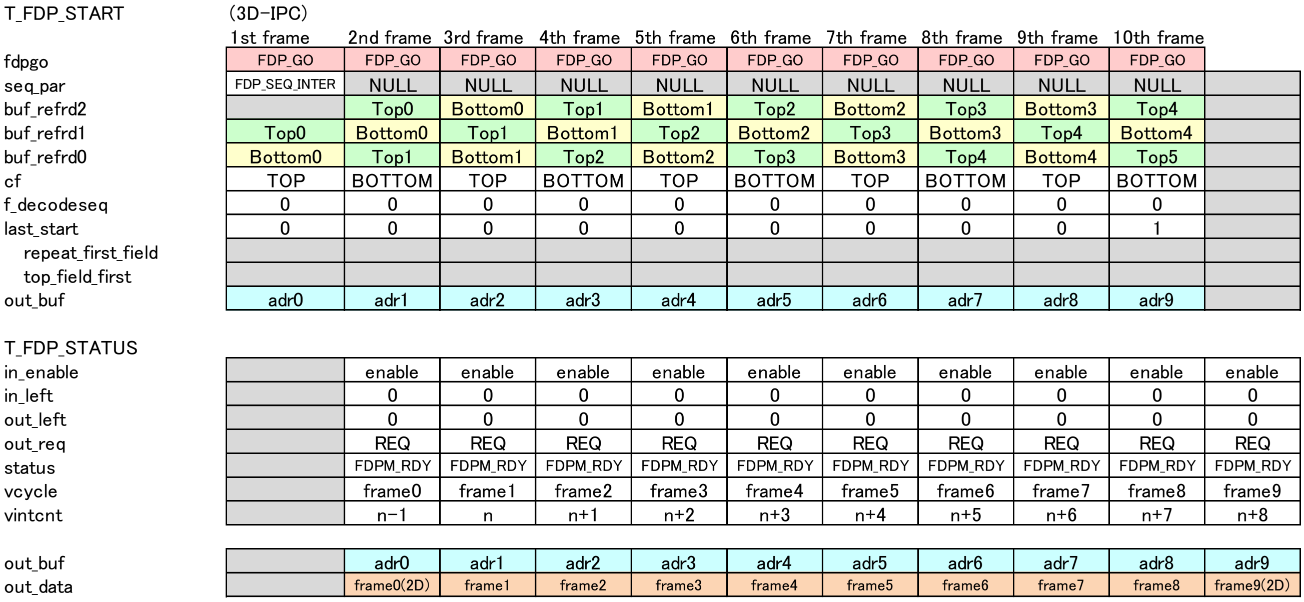
3.3.3 Sequence

In this chapter, show FDP Manager API arguments setting example in case of each sequence. In following example, Principal parameter describe in table 3.4. T_FDP_STATUS value shows under next T_FDP_START value. This value valid from current drv_FDPM_Start function to next drv_FDPM_Start.(example, 2nd frame column's T_FDP_STATUS value validate from 1st frame drv_FDP_Start to 2nd frame drv_FDP_Start).

table 3.4 principal parameter

T_FDP_START		
valiable name	description	member
fdpgo	FDP GO/NOGO	
seq_par	sequence parameter	fproc_par
buf_refrd2	Previous field address	fproc_par->ref_buf
buf_refrd1	Current field address	fproc_par->ref_buf
buf_refrd0	Next field address	fproc_par->ref_buf
cf	current field	fproc_par
f_decodeseq	force decode sequence	fproc_par
last_start	last start sequence	fproc_par
progressive_sequence	decode information	fproc_par->in_pic->pic_par
progressive_frame	decode information	fproc_par->in_pic->pic_par
repeat_first_field	decode information	fproc_par->in_pic->pic_par
top_field_first	decode information	fproc_par->in_pic->pic_par
out_buf	Output address	fproc_par

(1) 3D-IPC



T_FDP_START	(3D-IPC) 1st frame	2nd frame	3rd frame	4th frame	5th frame	6th frame	7th frame	8th frame	9th frame	10th frame	
fdpgo	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	
seq_par	FDP_SEQ_INTER	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	
buf_refrd2		Top0	Bottom0	Top1	Bottom1	Top2	Bottom2	Top3	Bottom3	Top4	
buf_refrd1	Top0	Bottom0	Top1	Bottom1	Top2	Bottom2	Top3	Bottom3	Top4	Bottom4	
buf_refrd0	Bottom0	Top1	Bottom1	Top2	Bottom2	Top3	Bottom3	Top4	Bottom4	Top5	
cf	TOP	BOTTOM	TOP	BOTTOM	TOP	BOTTOM	TOP	BOTTOM	TOP	BOTTOM	
f_decodeseq	0	0	0	0	0	0	0	0	0	0	
last_start	0	0	0	0	0	0	0	0	0	1	
repeat_first_field											
top_field_first											
out_buf	adr0	adr1	adr2	adr3	adr4	adr5	adr6	adr7	adr8	adr9	
T_FDP_STATUS											
in_enable		enable	enable	enable	enable	enable	enable	enable	enable	enable	enable
in_left		0	0	0	0	0	0	0	0	0	0
out_left		0	0	0	0	0	0	0	0	0	0
out_req		REQ	REQ	REQ	REQ	REQ	REQ	REQ	REQ	REQ	REQ
status		FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY
vcycle		frame0	frame1	frame2	frame3	frame4	frame5	frame6	frame7	frame8	frame9
vintcnt		n−1	n	n+1	n+2	n+3	n+4	n+5	n+6	n+7	n+8
out_buf		adr0	adr1	adr2	adr3	adr4	adr5	adr6	adr7	adr8	adr9
out_data		frame0(2D)	frame1	frame2	frame3	frame4	frame5	frame6	frame7	frame8	frame9(2D)

Figure 3-7 example setting in case of 3D-IPC sequence

In 3D-IPC sequence, 1st drv_FDP_Start call with valid seq_par (seq_mode=FDP_SEQ_INTER) and current frame field order cf(Top or Bottom). Following frames drv_FDP_Start call with seq_par =NULL. FDP Manager process as 2D-IPC for 1st frame automatically, and following frames process as 3D-IPC. Set buf_refrd2 for previous field, buf_refrd1 for current field and buf_refrd0 for next field. At last frame, drv_FDP_Start call with last_start=1. FDP Manager process as 2D-IPC for last frame.

(2) 2D-IPC

T_FDP_START	(2D-IPC) 1st frame	2nd frame	3rd frame	4th frame	5th frame	6th frame	7th frame	8th frame	9th frame	10th frame	
fdpgo	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	
seq_par	FDP_SEQ_INTER_2D	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	
buf_refrd2											
buf_refrd1	Top0	Bottom0	Top1	Bottom1	Top2	Bottom2	Top3	Bottom3	Top4	Bottom4	
buf_refrd0											
cf	TOP	BOTTOM	TOP	BOTTOM	TOP	BOTTOM	TOP	BOTTOM	TOP	BOTTOM	
f_decodeseq	0	0	0	0	0	0	0	0	0	0	
last_start	0	0	0	0	0	0	0	0	0	0	
repeat_first_field											
top_field_first											
out_buf	adr0	adr1	adr2	adr3	adr4	adr5	adr6	adr7	adr8	adr9	

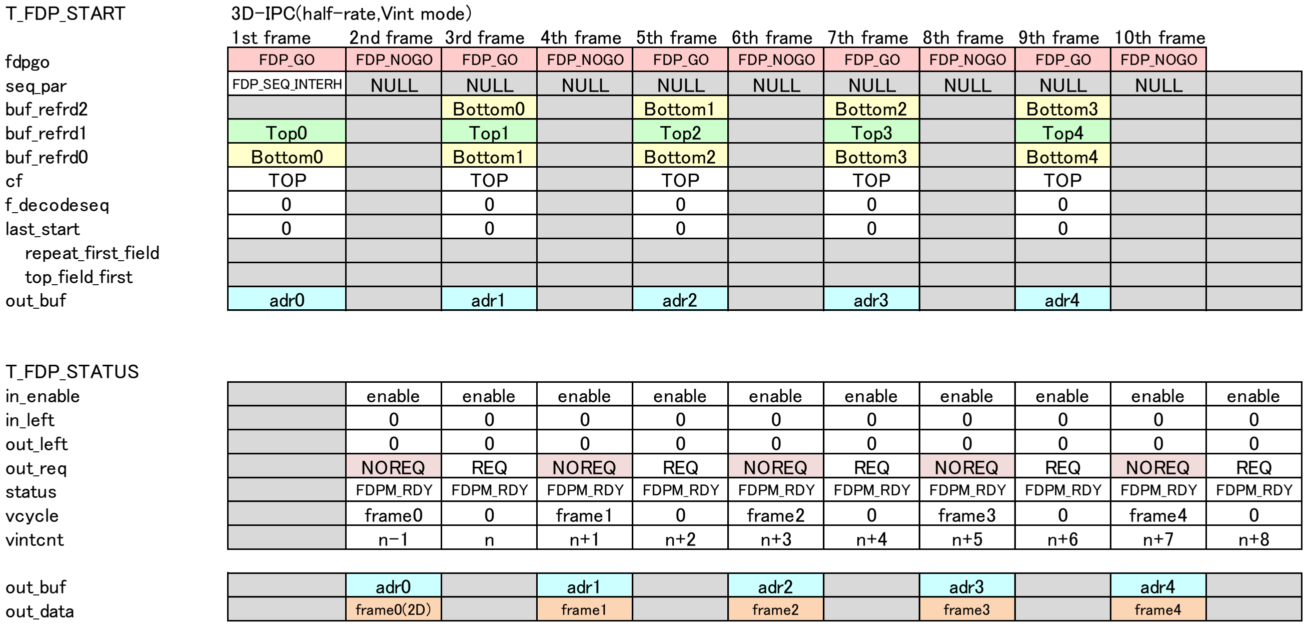
T_FDP_STATUS											
in_enable		enable	enable	enable	enable	enable	enable	enable	enable	enable	enable
in_left		0	0	0	0	0	0	0	0	0	0
out_left		0	0	0	0	0	0	0	0	0	0
out_req		REQ	REQ	REQ	REQ	REQ	REQ	REQ	REQ	REQ	REQ
status		FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY
vcycle		frame0	frame1	frame2	frame3	frame4	frame5	frame6	frame7	frame8	frame9
vintcnt		n-1	n	n+1	n+2	n+3	n+4	n+5	n+6	n+7	n+8

out_buf		adr0	adr1	adr2	adr3	adr4	adr5	adr6	adr7	adr8	adr9
out_data		frame0(2D)	frame1(2D)	frame2(2D)	frame3(2D)	frame4(2D)	frame5(2D)	frame6(2D)	frame7(2D)	frame8(2D)	frame9(2D)

Figure 3-8 example setting in case of 2D-IPC sequence

In 2D-IPC sequence, 1st drv_FDP_Start call with valid seq_pat(seq_mode=FDP_SEQ_INTER_2D) and current frame field order cf(Top or Bottom). Following frames drv_FDP_Start call with seq_par=NULL. Set buf_refrd1 for current field input.

(3) 3D-IPC(half rate, Vint mode)



3D-IPC(half-rate,Vint mode)

T_FDP_START	1st frame	2nd frame	3rd frame	4th frame	5th frame	6th frame	7th frame	8th frame	9th frame	10th frame	
fdpgo	FDP_GO	FDP_NOGO	FDP_GO	FDP_NOGO	FDP_GO	FDP_NOGO	FDP_GO	FDP_NOGO	FDP_GO	FDP_NOGO	
seq_par	FDP_SEQ_INTERH	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	
buf_refrd2			Bottom0		Bottom1		Bottom2		Bottom3		
buf_refrd1	Top0		Top1		Top2		Top3		Top4		
buf_refrd0	Bottom0		Bottom1		Bottom2		Bottom3		Bottom4		
cf	TOP		TOP		TOP		TOP		TOP		
f_decodeseq	0		0		0		0		0		
last_start	0		0		0		0		0		
repeat_first_field											
top_field_first											
out_buf	adr0		adr1		adr2		adr3		adr4		

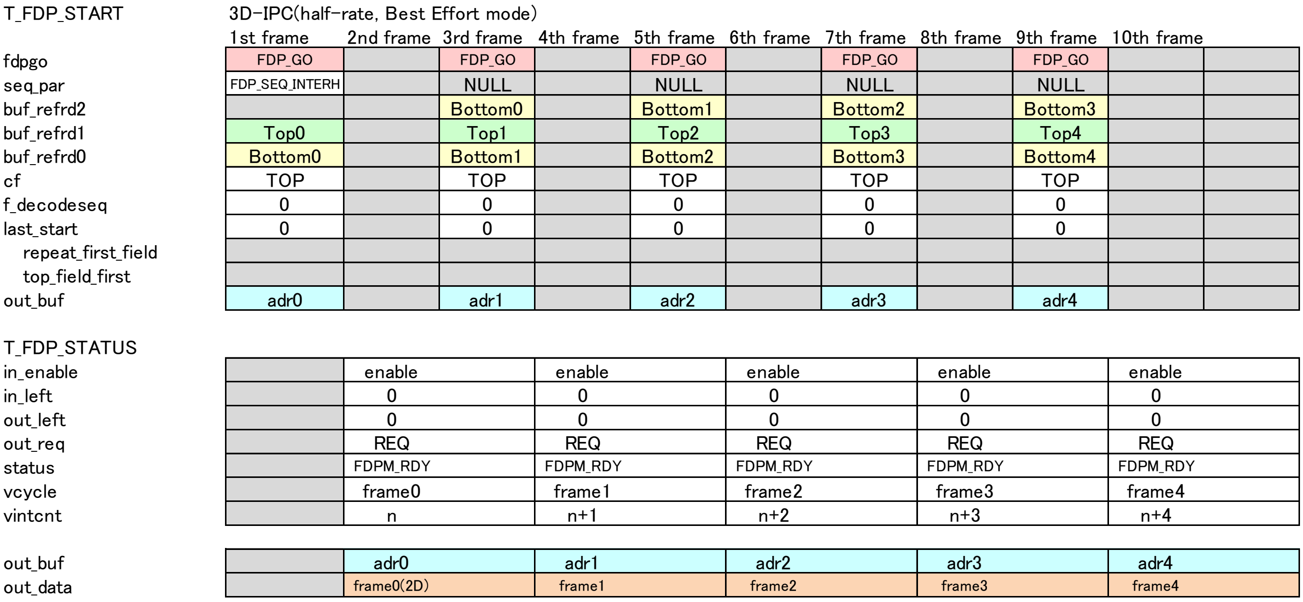
T_FDP_STATUS											
in_enable		enable	enable	enable	enable	enable	enable	enable	enable	enable	enable
in_left		0	0	0	0	0	0	0	0	0	0
out_left		0	0	0	0	0	0	0	0	0	0
out_req		NOREQ	REQ	NOREQ	REQ	NOREQ	REQ	NOREQ	REQ	NOREQ	REQ
status		FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY
vcycle		frame0	0	frame1	0	frame2	0	frame3	0	frame4	0
vintcnt		n−1	n	n+1	n+2	n+3	n+4	n+5	n+6	n+7	n+8

out_buf		adr0		adr1		adr2		adr3		adr4	
out_data		frame0(2D)		frame1		frame2		frame3		frame4	

Figure 3-9 example setting in case of 3D-IPC half rate, Vint mode sequence

In half rate sequence (specify seq_mode = FDP_SEQ_INTERH) with vint mode, when drv_FDPM_Start with fdp_go=FDP_GO call, FDP Manager return next status as out_req=NOREQ. User application call next drv_FDPM_Start with fdp_go=FDP_NOGO. FDP Manager doesn't IP conversion this frame and return next status as out_req=REQ. these procedures continue each frame.

(4) 3D-IPC(half rate, Best Effort mode)

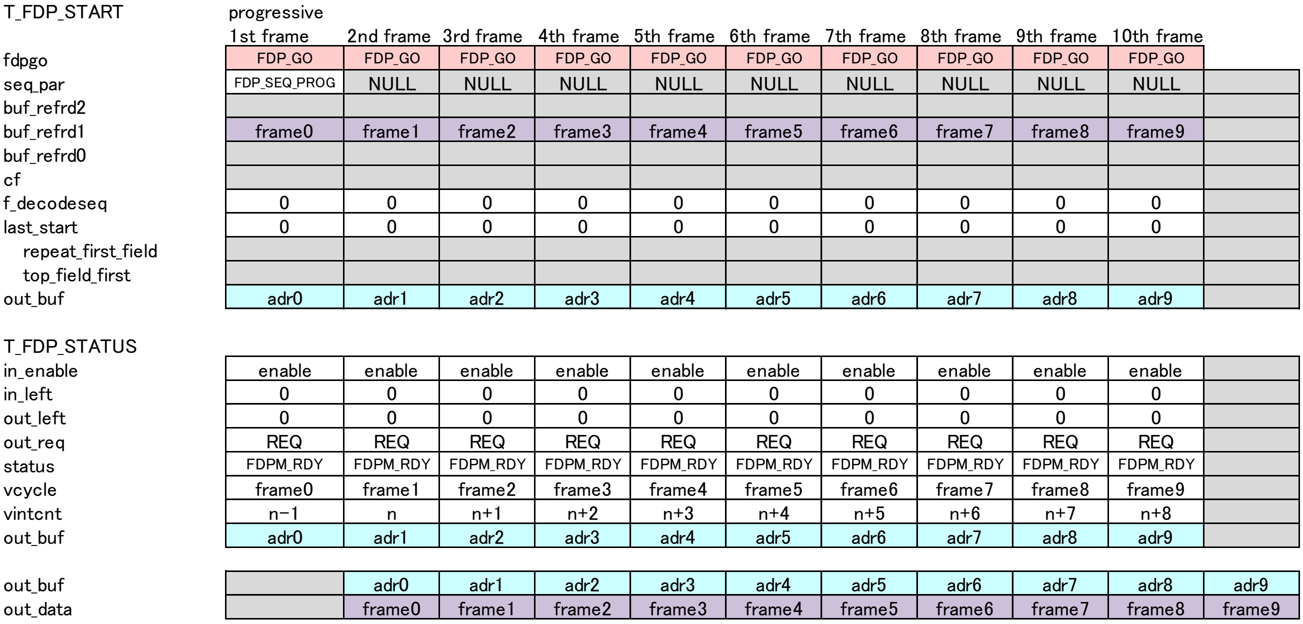


	1st frame	2nd frame	3rd frame	4th frame	5th frame	6th frame	7th frame	8th frame	9th frame	10th frame	
T_FDP_START	3D-IPC(half-rate, Best Effort mode)										
fdpgo	FDP_GO		FDP_GO		FDP_GO		FDP_GO		FDP_GO		
seq_par	FDP_SEQ_INTERH		NULL		NULL		NULL		NULL		
buf_refrd2			Bottom0		Bottom1		Bottom2		Bottom3		
buf_refrd1	Top0		Top1		Top2		Top3		Top4		
buf_refrd0	Bottom0		Bottom1		Bottom2		Bottom3		Bottom4		
cf	TOP		TOP		TOP		TOP		TOP		
f_decodeseq	0		0		0		0		0		
last_start	0		0		0		0		0		
repeat_first_field											
top_field_first											
out_buf	adr0		adr1		adr2		adr3		adr4		
T_FDP_STATUS											
in_enable		enable		enable		enable		enable		enable	
in_left		0		0		0		0		0	
out_left		0		0		0		0		0	
out_req		REQ		REQ		REQ		REQ		REQ	
status		FDPM_RDY		FDPM_RDY		FDPM_RDY		FDPM_RDY		FDPM_RDY	
vcycle		frame0		frame1		frame2		frame3		frame4	
vintcnt		n		n+1		n+2		n+3		n+4	
out_buf		adr0		adr1		adr2		adr3		adr4	
out_data		frame0(2D)		frame1		frame2		frame3		frame4	

Figure 3-10 example setting in case of 3D-IPC half rate, best effort mode sequence

In half rate sequence (specify seq_mode=FDP_SEQ_INTERH) with best effort mode, half rate is achieved by user application calling drv_FDPM_Start only one side.

(5) Progressive sequence



	1st frame	2nd frame	3rd frame	4th frame	5th frame	6th frame	7th frame	8th frame	9th frame	10th frame	
T_FDP_START	progressive										
fdpgo	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	
seq_par	FDP_SEQ_PROG	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	
buf_refrd2											
buf_refrd1	frame0	frame1	frame2	frame3	frame4	frame5	frame6	frame7	frame8	frame9	
buf_refrd0											
cf											
f_decodeseq	0	0	0	0	0	0	0	0	0	0	
last_start	0	0	0	0	0	0	0	0	0	0	
repeat_first_field											
top_field_first											
out_buf	adr0	adr1	adr2	adr3	adr4	adr5	adr6	adr7	adr8	adr9	
T_FDP_STATUS											
in_enable	enable	enable	enable	enable	enable	enable	enable	enable	enable	enable	
in_left	0	0	0	0	0	0	0	0	0	0	
out_left	0	0	0	0	0	0	0	0	0	0	
out_req	REQ	REQ	REQ	REQ	REQ	REQ	REQ	REQ	REQ	REQ	
status	FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY	FDPM_RDY	
vcycle	frame0	frame1	frame2	frame3	frame4	frame5	frame6	frame7	frame8	frame9	
vintcnt	n−1	n	n+1	n+2	n+3	n+4	n+5	n+6	n+7	n+8	
out_buf	adr0	adr1	adr2	adr3	adr4	adr5	adr6	adr7	adr8	adr9	
out_buf		adr0	adr1	adr2	adr3	adr4	adr5	adr6	adr7	adr8	adr9
out_data		frame0	frame1	frame2	frame3	frame4	frame5	frame6	frame7	frame8	frame9

Figure 3-11 example setting in case of progressive sequence

In progressive sequence, 1st drv_FDPM_Start with valid seq_par(seq_mode=FDP_SEQ_PROG). Following frames drv_FDPM_Start call with seq_par=NULL. Set buf_refrd1 for current frame input.

(6) 2:3 pull-down sequence (interlace sequence)

pull-down(interlace sequence)

	1st frame	2nd frame	3rd frame	4th frame	5th frame	6th frame	7th frame	8th frame	9th frame	10th frame	
T_FDP_START	Top0 Bottom0			Bottom1 Top1		Bottom2 Top2			Top3 Bottom3		
fdpgo	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	
seq_par	FDP_SEQ_INTER	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	
buf_refrd2		Top0	Bottom0		Top1		Top2	Bottom2		Top3	
buf_refrd1	Top0	Bottom0	Top0	Top1	Bottom1	Top2	Bottom2	Top2	Top3	Bottom3	
buf_refrd0	Bottom0	Top0		Bottom1		Bottom2	Top2		Bottom3		
cf	TOP	BOTTOM	TOP	BOTTOM	TOP	BOTTOM	TOP	BOTTOM	TOP	BOTTOM	
f_decodeseq	1	1	1	1	1	1	1	1	1	1	
last_start	0	0	0	0	0	0	0	0	0	0	
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	
out_buf	adr0	adr1	adr2	adr3	adr4	adr5	adr6	adr7	adr8	adr9	
out_buf		adr0	adr1	adr2	adr3	adr4	adr5	adr6	adr7	adr8	adr9
out_data		frame0	frame0	frame0	frame1	frame1	frame2	frame2	frame2	frame3	frame3

Figure 3-12 example setting in case of 2:3 pull-down interlace sequence

In case of pull-down sequence, user application set f_decodeseq=1 and set appropriate decode information (progressive_sequence, progressive_frame, picture_structure, repeat_first_field, top_field_first) and buf_refrd0-2.

(7) 2:3 pull-down sequence (progressive sequence)

pull-down(progressive sequence)

	1st frame	2nd frame	3rd frame	4th frame	5th frame	6th frame	7th frame	8th frame	9th frame	10th frame	
T_FDP_START	frame0		frame1			frame2		frame3			
fdpgo	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	
seq_par	FDP_SEQ_PROG	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	
buf_refrd2											
buf_refrd1	frame0	frame0	frame1	frame1	frame1	frame2	frame2	frame3	frame3	frame3	
buf_refrd0											
cf											
f_decodeseq	1	1	1	1	1	1	1	1	1	1	
last_start	0	0	0	0	0	0	0	0	0	0	
progressive_sequnece	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	
out_buf	adr0	adr1	adr2	adr3	adr4	adr5	adr6	adr7	adr8	adr9	
out_buf		adr0	adr1	adr2	adr3	adr4	adr5	adr6	adr7	adr8	adr9
out_data		frame0	frame0	frame1	frame1	frame1	frame2	frame2	frame3	frame3	frame3

Figure 3-13 example setting in case of 2:3 pull-down progressive sequence

In case of pull-down progressive sequence, user application set f_decodeseq=1 and set appropriate decode information (progressive_sequence, progressive_frame, picture_structure, repeat_first_field, top_field_first) and buf_refrd0.

3.3.4 Film detection

FDP Manager has film detection function. This function can use in case of 3D-IPC full rate sequence mode. When FDP Manager recognize as film, FDP Manager change from 3D-IPC mode to recombine the fields together back into original full frames. And when FDP Manager recognize as video, FDP Manager return to 3D-IPC mode. If user application uses film detection function, telecine_mode parameter set to FDP_TC_ON in start parameter (start_par->fproc_par->seq_par->telecine_mode). In this mode, user application input sequences same as video sequence.

3.3.5 Callback timing

FDP Manager have four kind of callback (callback1 ~ callback4).

Callback1 is set in drv_FDPM_Start. Callback1 call timing is after FDP Manager entry request start IPC to FDP Manager Kernel layer.

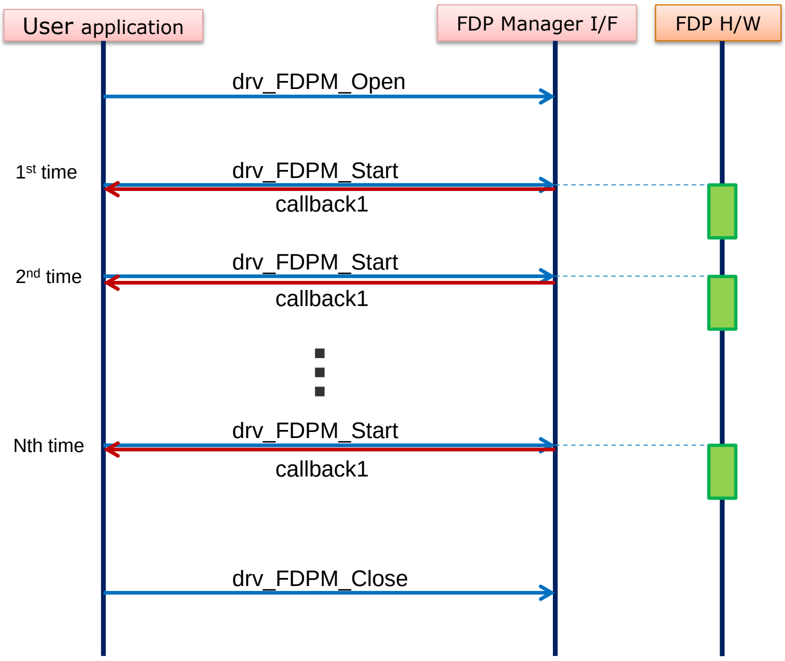


Figure 3-14 callback1 timing

Callback2 is set in drv_FDPM_Open(Vint mode application only) and drv_FDPM_Start. In Vint mode application, if drv_FDPM_Start called in Vperiod. Callback2 called. If drv_FDPM_Start does not call in Veriod, Callback2 does not call except first Vperiod timing after drv_FDPM_Open called. Callback2 call timing is after Vperiod time. If FDP H/W does not finished at Vperiod time, Callback2 timing delay to FDP H/W finish timing. In Best Effort mode application, callback2 call timing is at FDP H/W finished processing IPC.

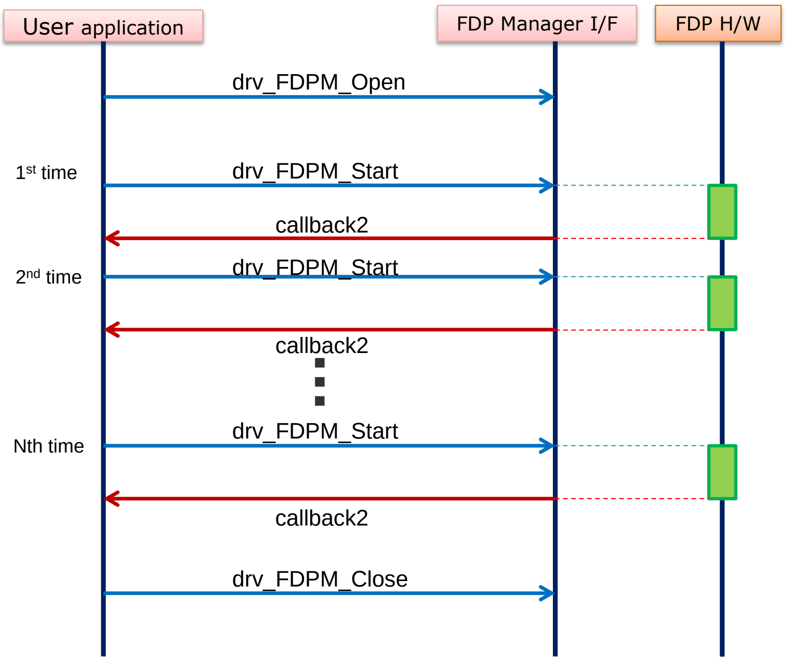


Figure 3-15 callback2 timing in best effort mode

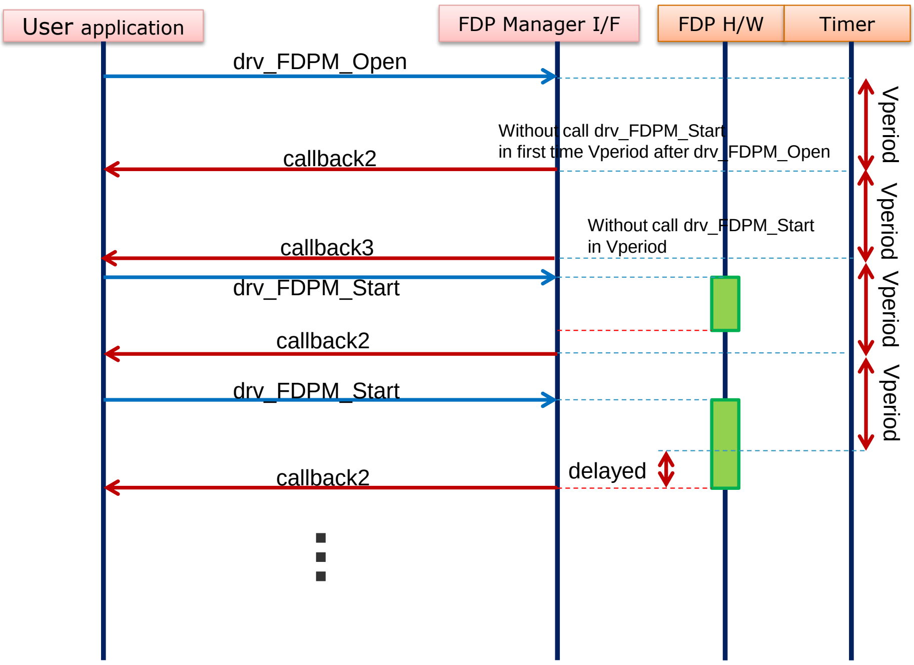


Figure 3-16 callback2/callback3 timing in Vint mode

Callback3 is set in drv_FDPM_Open(Vint mode application only). If drv_FDPM_Start does not call in Vperiod, callback3 is called. Callback3 call timing is after Vperiod time. If drv_FDPM_Start called in Vperiod, Callback3 does not call.

Callback4 is set in drv_FDPM_Open. Callback4 call timing is after time out period (default 1second) without call drv_FDPM_Start. Time out timer start at drv_FDPM_Open, and extended 1second at drv_FDPM_Start called timing. So, if drv_FDPM_Open call, you should call drv_FDPM_Start within time out period(1sec).

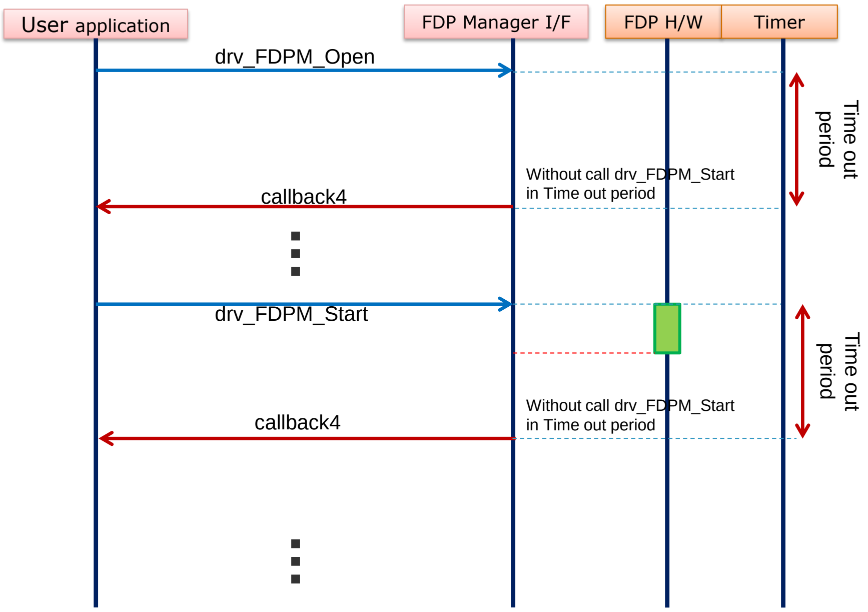


Figure 3-17 callback4 timing

4. List of API

FDP Manager I/F API display in Table 4.1.

Table 4.1 FDP Manager I/F API

Name	Function
drv_FDPM_Open	Request get resource to FDP Manager
drv_FDPM_Close	Request release resource to FDP Manager
drv_FDPM_Start	Request start FDP processing
drv_FDPM_Cancel	Request cancel FDP processing
drv_FDPM_Status	Get status of FDP processing

5.API Specification

Figure 5-1 shows the basic processing procedure using this software. The FDP Manager I/F have complex processing to communicate callback. However, this figure was simplifying the arrow.

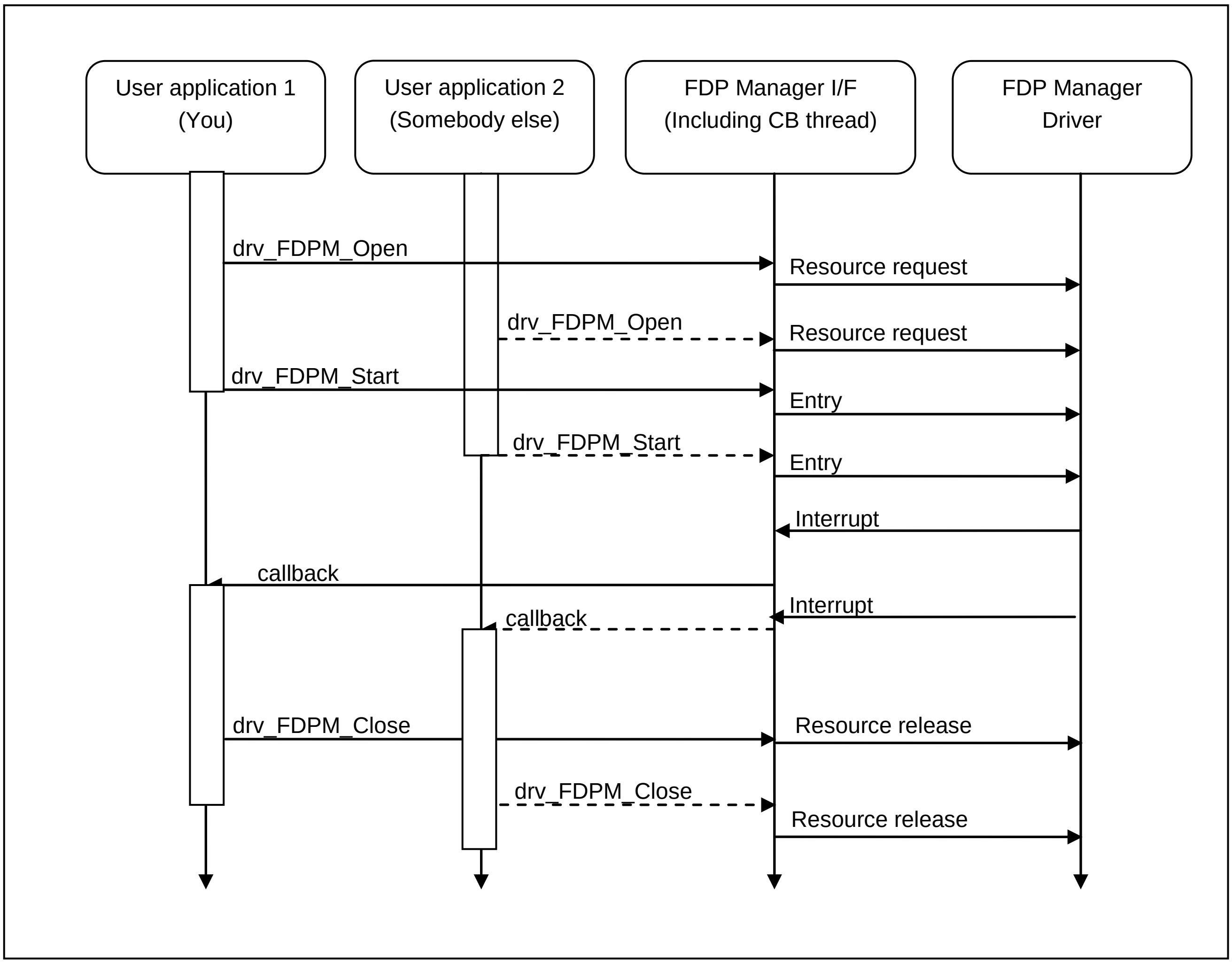


Figure 5-1 basic processing procedure

5.1 State chart and State Matrix

The state is managed in each ID allocated by drv_FDPM_Open().

The number of the states is 2.

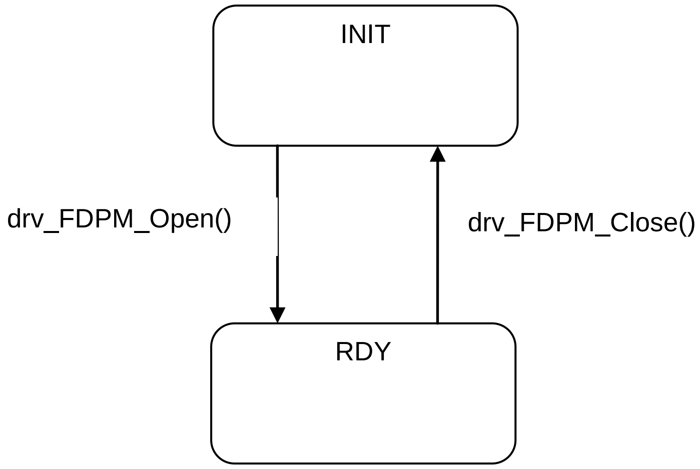
INIT

INIT is the state before FDP Manager I/F is opened.

RDY

RDY is the state after get FDP H/W resource by drv_FDPM_Open().

Application can start IP conversion process in this state.



State API	INIT	RDY
drv_FDPM_Open()	Permit →RDY	Permit(*1)
drv_FDPM_Close()	N/A	Permit →INIT(*2)
drv_FDPM_Start()	N/A	Permit
drv_FDPM_Cancel()	N/A	Permit
drv_FDPM_Status()	Permit	Permit

(*1)Application with ocmode=FDP_OCMODE_OCCUPY and vmode=FDP_VMODE_VBEST /FDP_VMODE_VBEST_FDP0/1/2 in T_FDP_OPEN permit. Other application not available.
(*2) last call will be move state to INIT.

5.2 Detail of FDP Manager I/F API

5.2.1 drv_FDPM_Open

Name

drv_FDPM_Open

Synopsis

errno drv_FDPM_Open(void *callback2, void *callback3, void *callback4, T_FDP_OPEN *open_par, FP user_function, void *userdata2, void *userdata3, void *userdata4, int *sub_ercd);

Arguments

void *callback2 :function pointer to callback function at V-sync interrupt timing.

void *callback3 :function pointer to callback function at V-sync interrupt with no Start function call.

void *callback4 :function pointer to callback function at time out detect.

T_FDP_OPEN *open_par : pointer to open parameter

FP user_function : function pointer to user_function

void *userdata2 :pointer to user data in callback2

void *userdata3 :pointer to user data in callback3

void *userdata4 :pointer to user data in callback4

int *sub_ercd : In case of occupy mode and Best effort mode application, pointer to FDPM handle.(if drv_FDPM_Open return value 0, FDP Manager set FDPM handle). In other application, pointer to sub error code.

Return value

0: success

EINVAL : parameter error. Refer to sub error code.

EACCES : error occurred due to call this function in invalid state.

< sub error code >

E_FDP_PARA_CB2	: invalid callback2
E_FDP_PARA_CB3	: invalid callback3
E_FDP_PARA_CB4	: invalid callback4
E_FDP_PARA_OPENPAR	: invalid open_par parameter.
E_FDP_PARA_REFBUFMODE	: invalid refbuf_mode parameter.
E_FDP_PARA_REFBUF	: invalid refbuf parameter.
E_FDP_PARA_VMODE	: invalid vmode parameter.

E_FDP_PARA_OCMODE	: invalid ocmode parameter.
E_FDP_PARA_INSIZE	: invalid insize parameter.
E_FDP_PARA_VCNT	: invalid vcnt parameter.

Description

Request reserve FDP resource.

FDP Manager check requested mode (FDP occupy/common user mode, Vint mode/Best Effort mode) in specified open parameter, if can get FDP resource, continue following process and return 0 error code. If cannot get FDP resource, return error code and finish this function.

Execute user function specified in argument user_function.

(1) In case of Vint mode

FDP Manager setting POSIX timer, register callback function (callback2). FDP Manager execute callback2 only 1time at following V-sync timing. Register callback function (callback3). FDP Manager executes callback3 at V-sync timing with no Start function call.

(2) In case of Best Effort mode

Ignore callback2, callback3 setting.(in case of callback at FDP processing finish interrupt, specify in drv_FDPM_Start function)

FDP Manager registers callback function (callback4). FDP Manager executes callback4 at time out.

Table 5.1 member of drv_FDPM_Open

Type/member	Input/Output	Description
void *callback2	Input	Function pointer to callback function at V-sync interrupt timing. In case of Best Effort mode, specify NULL. Other case, do not specify NULL. About Callback function spec, refer to drv_FDPM_Start function.
void *callback3	Input	Function pointer to callback function at V-sync interrupt timing with no Start function call. In case of Best Effort mode, specify NULL. About callback function spec, refer to drv_FDPM_Start function.
void *callback4	Input	Function pointer to callback function at time out. Do noy specify NULL. About callback function spec, refer to drv_FDPM_Start function.
T_FDP_OPEN *open_par	Input	Pointer to struct open parameter. Specify T_FDP_OPEN* type. Do not specify NULL
FP user_function	Input	Function pointer to user function. If specify NULL, FDP Manager do not execute user function. User function specification is below. Void user_function(void) <argument> none. <return value> none.
void *userdata2	Input	Pointer to user data for callback2

		If specify NULL, FDP manager do not set callback user data pointer.
void *userdata3	Input	Pointer to user data for callback3 If specify NULL, FDP manager do not set callback user data pointer.
void *userdata4	Input	Pointer to user data for callback4 If specify NULL, FDP manager do not set callback user data pointer.
int *sub_ercd	Output	Pointer to FDPM handle (in case of best effort mode application) FDP Manager set FDPM handle to this argument. If user application use best effort mode, do not specify NULL. Pointer to sub error code(in case of vint mode application) In this mode, If specify NULL, FDP manager do not return sub error code.

Notes

In case of re-call this function in thread, execute after drv_FDPM_Close.

See Also

None.

5.2.2 drv_FDPM_Close

Name

drv_FDPM_Close

Synopsis

errno drv_FDPM_Close(FP user_function, int *sub_ercd, unsigned cahr f_release);

Arguments

FP user_function : function pointer to user function.

int *sub_ercd : In case of occupy mode and Best effort mode application, pointer to FDPM handle. set to FDPM handle (return sub_ercd when call drv_FDPM_open). In other application, pointer to sub error code.

unsigned char f_release :force release setting

Return value

0: success

EACCES : error occurred due to call this function in invalid state.

Description

Close FDP processing. Release FDP resource. In case of Vint mode, stop timer functions. Execute user function specified in argument user_function.

If this function call with force close mode (f_release=1), FDP Manager close process by force (whether remain processing request or not). If this function call with normal close mode (f_release=0), in case of remaining processing request, FDP Manager return EACCESS (do not execute close process).

Table 5.2 member of drv_FDPM_Close

Type/member	Input/Output	Description
FP user_function	Input	Function pointer to user function If specify NULL, FDP Manager do not execute user function. User function specification is below. Void user_function(void) <argument> none. <return value> none.

int *sub_ercd	Input(best effort mode)
	Output(other mode)
		Pointer to FDPM handle(in case of best effort mode application)
		Set FDPM handle (return sub_ercd when call drv_FDPM_Open).
		do not specify NULL.
		In other mode application, Pointer to sub error code

unsigned char f_close	Input	Force close mode setting 1:force close mode 0:normal close mode

Notes

This function call after drv_FDPM_Open.(do not call this function without open operation)

See Also

None.

5.2.3 drv_FDPM_Start

Name

drv_FDPM_Start

Synopsis

errno drv_FDPM_Start(void *callback1, void *callback2, T_FDP_START *start_par, void *userdata1, void *userdata2, int *sub_ercd);

Arguments

void *callback1 :function pointer to callback function(callback1) at this driver call.

void *callback2 :function pointer to callback function(callback2) at V-sync interrupt timing

(in case of V-int mode) or FDP process finish timing(in case of Best effort mode).

T_FDP_START *start_par : pointer to start parameter.

void *userdata1 : pointer to user data in callback1 void *userdata2 : pointer to user data in callback2

int *sub_ercd :pointer to FDPM handle(best effort mode application), in case of best effort mode application, set to FDPM handle(return sub_ercd when call drv_FDPM_open). In case of other mode application, pointer to sub error code.

Return value

0	:success
EINVAL	: parameter error. Refer to sub error code.
EACCES	: error occurred due to call this function in invalid state.

[sub error code]

In case of return EINVAL

E_FDP_PARA_CB1	: invalid callback1
E_FDP_PARA_CB2	: invalid callback2
E_FDP_PARA_STARTPAR	: invalid start_par parameter.
E_FDP_PARA_VCNT	: invalid vcnt parameter
E_FDP_PARA_FDPGO	: invalid fdpgo parameter
E_FDP_PARA_FPROCPAR	: invalid fproc_par parameter
E_FDP_PARA_SEQPAR	: invalid seq_par parameter
E_FDP_PARA_INPIC	: invalid in_pic parameter
E_FDP_PARA_OUTBUF	: invalid out_buf parameter
E_FDP_PARA_REFBUF	: invalid ref_buf parameter
E_FDP_PARA_SEQMODE	: invalid seq_mode parameter
E_FDP_PARA_TELECINEMODE	: invalid telecine_mode parameter
E_FDP_PARA_INWIDTH	: invalid inwidth parameter
E_FDP_PARA_INHEIGHT	: invalid inheght parameter
E_FDP_PARA_PICPAR	: invalid pic_par parameter
E_FDP_PARA_PICWIDTH	: invalid width in pic_par parameter
E_FDP_PARA_PICHEIGHT	: invalid height in pic_par parameter
E_FDP_PARA_CHROMA	: invalid chroma_format parameter
E_FDP_PARA_PROGSEQ	: invalid progressive_sequence parameter

E_FDP_PARA_PICSTRUCT: invalid picture_structure parameter

E_FDP_PARA_REPEATTOP: invalid repeat_first_field and top_field_first parameter

E_FDP_PARA_BUFREFRD0 : invalid buf_refrd0 parameter

E_FDP_PARA_BUFREFRD1 : invalid buf_refrd1 parameter

E_FDP_PARA_BUFREFRD2 : invalid buf_refrd2 parameter

E_FDP_PARA_BUFADDR : invalid addr in T_FDP_IMGBUF parameter

E_FDP_PARA_BUFADDRC : invalid addr_c0 and addr_c1 in T_FDP_IMGBUF parameter

E_FDP_PARA_BUFSTRIDE : invalid stride in T_FDP_BUF parameter

E_FDP_PARA_BUFHEIGHT : invalid height in T_FDP_BUF parameter

E_FDP_PARA_BUFHEIGHT_C : invalid height_c in T_FDP_BUF parameter.

E_FDP_PARA_FIELD_PARITY :invalid field pality

Description

FDP start IP conversion processing. FDP Manager returns 0 and sub error code as accepted id number. After start function setting, execute callback function (callback1). FDP Manager registers callback2. In case of Vint mode, FDP Manager execute callback2 only 1time at following V-sync timing. In case of Best effort mode, FDP Manager executes callback2 only 1time at FDP process finish timing.

Table 5.3 member of drv_FDPM_Start

Type/member	Input/Output	Description
void *callback1	Input	Function pointer to callback1. Do not specify NULL Callback1 specification is below. *void callback1(T_FDP_CB1 *fdp_cb1);* <argument> *T_FDP_CB1 *fdp_cb1*: pointer to FDP processing information. <return value> none.
void *callback2	Input	Function pointer to callback2 Do not specify NULL. callback2 specification is below. *void callback2(T_FDP_CB2 *fdp_cb2);* <argument> T_FDP_CB2 *fdp_cb2:pointer to FDP processing finish information. <return value> none.
T_FDP_START *start_par	Input	Pointer to start parameter Do not specify NULL
void *usedata1	Input	Pointer to user data for callback function(callback1) If specify NULL, FDP manager do not set user data pointer for callback1 function.

void *userdata2	Input	Pointer to user data for callback function(callback2) If specify NULL, FDP manager do not set user data pointer for callback2 function.

int *sub_ercd	Input(best effort mode)
	Output(other mode)
		Pointer to FDPM handle (in case of best effort mode application)
		Set to FDPM handle(return sub_ercd when call drv_FDPM_Open).
		In case of other mode application, pointer to sub error code.
		If specify NULL FDP manager do not return sub error code.

FDP processing for decode information combination describe.

“X” indicates “don’t care”

Gray hatching indicates illegal combination for MPEG2.

Table 5.4 decode information combination (progressive_sequence=”1”)

< progressive sequence(progressive_sequence=”1”) >

Progressive_frame	Picture_structure	Repeat_first_field	Top_firld_first	process
“X”	“0b11”	“0b0”	“0b0”	Input 1time
		“0b0”	“0b1”	Error(E_FDP_PARA_REPETTOP)
		“0b1”	“0b0”	Repeat input 2times
		“0b1”	“0b1”	Repeat input 3times
	Other “0b11”	“X”	“X”	error

Table 5.5 decode information combination (progressive_sequence=”0”)

< interlace sequence (progressive_sequence=”0”) >

Progressive_frame	Picture_structure	Repeat_first_field	Top_field_first	Process
“X”	“0b00”	“X”	“X”	error
“0b0”	“0b01”	“X”	“X”	Input top field 1time
	“0b10”	“X”	“X”	Input bottom field 1time
	“0b11”	“0b0”	“0b0”	Input BOT->TOP order
		“0b0”	“0b1”	Input TOP->BOT order
		“0b1”	“0b0”	Input BOT->TOP->BOT order(*1)
		“0b1”	“0b1”	Input TOP->BOT-TOP order(*1)
“0b1”	“0b01”	“X”	“X”	Error
	“0b10”	“X”	“X”	Error
	“0b11”	“0b0”	“0b0”	Input BOT->TOP order
		“0b0”	“0b1”	Input TOP->BOT order
		“0b1”	“0b0”	Input BOT->TOP->BOT order
		“0b1”	“0b1”	Input TOP->BOT-

				>TOP order

(*1) This combination does not specify in MPEG2 standard.

Each field input manually. Each field input, need to set parameter follow chapter.

Notes

None.

See Also

None.

5.2.4 drv_FDPM_Cancel

Name

drv_FDPM_Cancel

Synopsis

errno drv_FDPM_Cancel(int id, int *sub_ercd);

Arguments

int id :cancel ID number

int *sub_ercd : pointer to FDPM handle(in case of best effort mode application). In case of other mode application, pointer to sub error code

Return value

0: success

EINVAL: parameter error refer to sub error code.

[Sub error code]

E_FDP_CANCEL_NOID: no such ID process or already finished.

E_FDP_CANCEL_ID_PROCESSING: running process ID (Cannot cancel process)

Description

Specified ID cancel FDP processing request. FDP Manager delete specified ID’s request from FDP processing request queue. If already processing start, return EINVAL and sub error code.

Table 5.6 member of drv_FDPM_Cancel

Type / member	Input/Output	Description
int id	Input	Specify cancel ID

int *sub_ercd	Input(best effort mode)
	Output(other mode)
		Pointer to FDPM handle (in case of best effort mode application)
		Set to FDPM handle(return sub_ercd when call drv_FDPM_Open).
		In case of other mode application, pointer to sub error code.
		If specify NULL FDP manager do not return sub error code.

Note

None.

See Also

None.

5.2.5 drv_FDPM_Status

Name

drv_FDPM_Status

Synopsis

errno drv_FDPM_Status(T_FDP_STATUS *fdp_status, int *sub_ercd);

Arguments

T_FDP_STATUS *fdp_status: pointer to FDP processing status information

int *sub_ercd: pointer to FDPM handle (in case of best effort mode application). In other mode application, pointer to sub error code

Return value

0: success

EINVAL: parameter error. Refer to sub error code.

[Sub error code]

E_FDP_PARA_STATUS invalid fdp_status

Description

This function return FDP processing status information to fdp_status. Allocate memory greater than struct T_FDP_STATUS size.

Table 5.7 member of drv_FDPM_Status

Type / member	Input/output	Description
T_FDP_STATUS *fdp_status	Output	Pointer to FDP status information Do not specify NULL

int *sub_ercd	Input(best effort mode)
	Output(other mode)
		Pointer to FDPM handle (in case of best effort mode application)
		Set to FDPM handle(return sub_ercd when call drv_FDPM_Open).
		In case of other mode application, pointer to sub error code.
		If specify NULL FDP manager do not return sub error code.

Notes

None.

See Also

None.

6. FDP Manager arguments

6.1 T_FDP_OPEN

The following is described about the member of T_FDP_OPEN structure.

```
typedef struct{
        unsigned char       ref_mode;
        unsigned char       refbuf_mode;
        T_FDP_REFPREBUF          *refbuf;
        unsigned char       ocmode;
        unsigned char       vmode;
        T_FDP_IMGSIZE   *insize;
        unsigned char       clkmode;
        unsigned long       vcnt;
} T_FDP_OPEN;
```

Table 6.1 member of T_FDP_OPEN

Type/member	Input/output	Description
unsigned char ref_mode	input	Reference picture mode Specify “0”
unsigned char refbuf_mode	input	Reference picture buffer mode Specify “0”
T_FDP_REFPREBUF *refbuf	input	Pointer to pre-reserve reference picture buffer Specify NULL
unsigned char ocmode	input	FDP occupy mode setting FDP_OCMODE_OCCUPY:occupy mode FDP_OCMODE_COMMON:common use mode
unsigned char vmode	input	V-sync mode setting FDP_VMODE_NORMAL:Vint mode FDP_VMODE_VBEST:Best Effort mode FDP_VMODE_VBEST_FDP0:Best Effort mode(using FDP0)(*1) FDP_VMODE_VBEST_FDP1:Best Effort mode(using FDP1)(*1) FDP_VMODE_VBEST_FDP2:Best Effort mode(using FDP2)(*1),(*2)
T_FDP_IMGSIZE *insize	input	Pointer to Input picture size Setting pointer to T_FDP_IMGSIZE which specify input picture size. Struct T_FDP_IMGSIZE member: Inhsize: input horizontal size 80-1920pixel only even number permit. Invsize:input vertical size In case of interlace sequence, 40-960pixel permit. In case of progressice sequence, 80-1920pixel only even number permit. FDP Manager allocates memory for FDP H/W internal use based on picture size in this parameter. Set maximum size input picture size from drv_FDPM_Open to drv-FDPM_Close.
unsigned char clkmode	input	Clock stop mode Specify “FDP_CLKMODE_1”

unsigned long vcnt	input	V-sync interrupt timing 0.1ms order set as 0.1ms=1 Example:16.6ms = 166

(*1) These mode are specify using FDP H/W. if specified FDP H/W are used from other Vint mode application, drv_FDPM_Open function return with –EINVAL.

(*2)This mode permit using on R-CarH2 only. In case of R-CarM2, do not set this mode.

6.2 T_FDP_IMGSIZE

The following is described about the member of T_FDP_IMGSIZE structure.

```
typedef struct{
        unsigned short width;
        unsigned short height;
} T_FDP_IMGSIZE;
```

Table 6.2 member of T_FDP_IMGSIZE

Type/member	Input/output	Description
unsigned short width	output	Picture horizontal size(pixel)
unsigned short height	output	Picture vertical size(pixel) In case of YUV420, number of C line is half of this value.

6.3 T_FDP_CB1

The following is described about the member of T_FDP_CB1 structure.

```
typedef struct{
        T_FDP_IMGBUF *buf_in;
        T_FDP_IMGSIZE *insize;
        T_FDP_IMGSIZE *refsize;
        T_FDP_IMGSIZE *iirsize;
        T_FDP_IMGSIZE *outsize;
        unsigned char refwr_num;
        unsigned char refrd0_num;
        unsigned char refrd1_num;
        unsigned char refrd2_num;
        unsigned char refwr_y_en;
        unsigned char refwr_c_en;
        unsigned char refrd0_en;
        unsigned char refrd1_en;
        unsigned char refrd2_en;
        unsigned char refiir_en;
        void *userdata1;
}T_FDP_CB1;
```

Table 6.3 member of T_FDP_CB1

Type/member	Input/output	Description
T_FDP_IMGBUF	output	NULL

*buf_in		
T_FDP_IMGSIZE *insize	output	Pointer to input image size if no input image, NULL
T_FDP_IMGSIZE *refsize	output	NULL
T_FDP_IMGSIZE *iirsize	output	NULL
T_FDP_IMGSIZE *outsize	output	Pointer to output image size If no output image, NULL
unsigned char refwr_num	output	Still mask write buffer number(0 or 1) This member only valid in case of refwr_y_en=1.
unsigned char refrd0_num	output	Not support
unsigned char refrd1_num	output	Not support
refrd2_num	output	Not support
unsigned char refwr_y_en	output	Still mask write enable 0:still mask do not write 1:still mask write
unsigned char refwr_c_en	output	Same as refwr_y_en
unsigned char refrd0_en	output	Refrd0 enable 0:refrd0 disable 1:refrd0 enable
unsigned char refrd1_en	output	Refrd1 enable 0:refrd1 disable 1:refrd1 enable
unsigned char refrd2_en	output	Refrd2 enable 0:refrd2 disable 1:refrd2 enable
unsigned char refiir_en	output	Allways “0”
void *userdata1	output	Pointer to userdata specified in userdata1

6.4 T_FDP_CB2

The following is described about the member of T_FDP_CB2 structure.

```
typedef struct{
        int ercd;
        void *userdata2;
} T_FDP_CB2;
```

Table 6.4 member of T_FDP_CB2

Type/member	Input/output	Description
int ercd	output	END status E_FDP_END:finish frame processing(include no processing) E_FDP_DELAYED:delay frame processing.
void *userdata2	output	Pointer to user data specified in userdata2

6.5 T_FDP_START

The following is described about the member of T_FDP_START structure.

```
typedef struct{
        unsigned long *vcnt;
        unsigned char fdpgo;
        T_FDP_FPROC     *fproc_par;
} T_FDP_START;
```

Table 6.5 member of T_FDP_START

Type/member	Input/output	description
unsigned long *vcnt	input	Pointer to Vperiod value Set Vperiod value as 0.1ms=1. Example: 16.6ms = 166 Range:160-700 If specify NULL, FDP Manger use previous setting (if first time set is NULL, use drv_FDPM_Open setting).
unsigned char fdpgo	input	Frame processing request FDP_NOGO:do not request(only update Vperiod) FDP_GO:request frame processing.
T_FDP_PROC *fproc_par	input	Pointer to frame processing parameter. This parameter valid in case of fdpgo=”FDP_GO”. In case of fdpgo=”FDP_GO”, do not specify NULL.

6.6 T_FDP_FPROC

The following is described about the member of T_FDP_FPROC structure.

```
typedef struct{
        T_FDP_SEQ       *seq_par,
        T_FDP_IMGET     *imgset_par;
        T_FDP_PIC       *in_pic;
        unsigned char   last_start;
        unsigned char   cf;
        unsigned char   f_decodeseq;
        T_FDP_IMGBUF    *out_buf;
        unsigned char   out_format;
        T_FDP_REFBUF    *ref_buf;
}T_FDP_FPROC;
```

Table 6.6 member of T_FDP_FPROC

Type/member	Input/output	Description
T_FDP_SEQ *seq_par	input	Pointer to sequence parameter. If specify NULL, FDP Manager use previous setting. If specify not NULL, FDP Manager recognize new sequence start. In case of fdp_status->in_enable=”FDP_IN_ENABLE” and fdp_status->in_left not equal 0, FDP manager can not start new sequence. In case of fdp_status->in_enable=”FDP_IN_ENABLE” and

		fdp_status->in_left qeual 0, FDP manager can start new sequence. In case of fdp_status->in_enable="FDP_IN_DISABLE", FDP manager can continue sequence when fdp_status->out_req="FDP_OUT_REQ". If start new sequence with fdp_status->seq_lock="FDP_SEQ_LOCK", FDP manager destruction previous input picture.
T_FDP_IMGSET *imgset_par	input	Ignore this parameter
T_FDP_PIC *in_pic	input	Pointer to struct of input picture. Do not specify NULL
unsigned char last_start	input	Last start indication 1: in case of 3D-IPC, in last frame prosessing, set 1 this member, FDP Manager force 2D mode processing. 0: normal processing. Except above case, set to 0.
unsigned char cf	input	Current field parity indication Set current field parity in case of interlace mode. In case of progressive mode, ignore this member. 0:Top field 1:bottom field
unsigned char f_decodeseq	input	Force decode information use mode. 1: force pull-down processing based on in_pic->pic_par decode information. 0:3D/2D IPC mode.
T_FDP_IMGBUF *out_buf	input	Pointer to struct of output buffer. If specify NULL, FDP manager do not output image. In case of progressive sequence and full rate processing, do not specify NULL. In case of half-rate processing, when fdp_status->out_req="FDP_OUT_REQ", do not specify NULL. When "FDP_OUT_NOREQ", specify NULL
unsigned char out_format	input	Output format FDP_YUV420:YUV420 semi-planar(NV12) FDP_YUV420_YV12:YUV420 planar(YV12) FDP_YUV420_NV21:YUV420 semi-planar(NV21) FDP_YUV_422_NV16:YUV422 semi-planar(NV16) FDP_YUV_422_YUY2:YUV422 packed(YUY2) FDP_YUV_422_UYVY:YUV422 packed(UYVY)
T_FDP_REFBUF *ref_buf	input	Pointer to struct of reference buffer. Do not specify NULL.

6.7 T_FDP_SEQ

The following is described about the member of T_FDP_SEQ structure.

```
typedef struct{
        unsigned char       seq_mode;
        unsigned char       scale_mode;
        unsigned char       filter_mode;
```

```
            unsigned char       telecine_mode;
            unsigned short      in_width;
            unsigned short      in_height;
            unsigned short      imgtop;
            unsigned short      imgleft;
            unsigned short      imgwidth;
            unsigned short      imgheight;
            unsigned short      out_width;
            unsigned short      out_height;
            T_FDP_RATIO         *ratio;
} T_FDP_SEQ;
```

Table 6.7 member of T_FDP_SEQ

Type/member	Input/output	Description
unsigned char seq_mode	input	Sequence mode FDP_SEQ_PROG:progressive FDP_SEQ_INTER:interlace full-rate output FDP_SEQ_INTERH:interlace half-rate output FDP_SEQ_INTER_2D:interlace full-rate output(2D-IPC) FDP_SEQ_INTERH_2D:interlace half-rate output(2D-IPC)
unsigned char scale_mode	input	Unused this setting.
unsigned char filter_mode	input	Unused this setting.
unsigned char telecine_mode	input	Telecine detect mode FDP_TC_OFF:telecine detect mode off FDP_TC_ON:telecine detect mode on. If use telecine detect mode on, input video sequence (fproc_par->f_decodeseq=0). If fproc_par->f_decodeseq=1, telecine detect mode disable.
unsigned short in_width	input	Input picture horizontal size 80-1920pixel even number only permit
unsigned short in_height	input	Input picture vertical size In interlace sequence, 40-960pixel permit. In progressive sequence, 80-1920pixel even number only permits.
unsigne short imgleft	input	Unused this member
unsigned short imgtop	input	Unused this member
unsigned short imgwidth	input	Unused this member
unsigned short imgheight	input	Unused this member
unsigned short out_width	Input	Unused this member
unsigned short out_height	input	Unused this member
T_FDP_RATIO *ratio	input	Set “NULL”

6.8 T_FDP_PIC

The following is described about the member of T_FDP_PIC structure.

```
typedef struct{
            unsigned long       picid;
            T_FDP_PICPAR        *pic_par;
            T_FDP_IMGBUF        *in_buf1;
            T_FDP_IMGBUF        *in_buf2;
} T_FDP_PIC;
```

Table 6.8 member of T_FDP_PIC

Type/member	Input/output	Description
unsigned long picid	input	Picture ID Set optional value. This value use identification each frame for user. This value reflects to in_picid and out_picid which are members of T_FDP_STATUS.
T_FDP_PICPAR *pic_par	input	Pointer to picture parameter Do not specify NULL
T_FDP_IMGBUF *in_buf1	input	Unused this member
T_FDP_IMGBUF *in_buf2	input	Unused this member

6.9 T_FDP_PICPAR

The following is described about the member of T_FDP_PICPAR structure.

```
typedef struct{
            unsigned short      width;
            unsigned short      height;
            unsigned char       chroma_format;
            unsigned char       progressive_sequence;
            unsigned char       progiressive_frame;
            unsigned char       picture_structure;
            unsigned char       repeat_first_field;
            unsigned char       top_field_first;
}T_FDP_PICPAR;
```

Table 6.9 member of T_FDP_PICPAR

Type/member	Input/output	Description
unsigned short width	input	Horizontal size Set same as start_par->fproc_par->seq_par->in_width
unsigned short height	input	Vertical size Set same as start_par->fproc_par->seq_par->in_height
unsigned char chroma_format	input	Input format FDP_YUV420:YUV420 semi-planar(NV12) FDP_YUV420_YV12:YUV420 planar(YV12) FDP_YUV420_NV21:YUV420 semi-planar(NV21) FDP_YUV_422_NV16:YUV422 semi-planar(NV16) FDP_YUV_422_YUY2:YUV422 packed(YUY2)

unsigned char progressice_sequence	input	Decode information Progressive_sequence Need consistency with seq_mode. (when seq_mode="FDP_SEQ_PROG", progressive_sequence=1, other case progressive_sequence=0)
unsigned char progressive_frame	input	Decode information progressive_frame
unsigned char picture_structure	Input	Decode information picture_structure
unsigned char repeat_first_field	Input	Decode information repeat_first_field
unsigned char top_field_first	Input	Decode information top_field_first

6.10 T_FDP_IMGBUF

The following is described about the member of T_FDP_IMGBUF structure.

```
typedef struct{
            void        *addr;
            void        *addr_c0;
            void        *addr_c1;
            unsigned short       stride;
            unsigned short      stride_c;
            unsigned short       height;
            unsigned short       height_c;
}T_FDP_IMGBUF;
```

Table 6.10 member of T_FDP_IMGBUF

Type/member	Input/output	Description
void *addr	input	Y buffer address Set Physical address with 1byte alignment. Do not specify NULL. YCbCr/Planer,Semi-Planer format: Y plane address YCbCr/Packed format: Y/Cb/Cr plane address
void *addr_c0	input	C0 buffer address Set physical address with 1byte alignment. When YCbCr/Planer or semi-planer format, do not specify NULL. YCbCr/Planar format: Cb plane address YCbCr/Semi-planer format: Cb/Cr plane address
void *addr_c1	input	C1 buffer address Set physical address with 1byte alignment. When YCbCr/Planar format, do not specify NULL.
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.
unsigned short height	input	Y buffer height Set greater than input picture vertical size.
unsigned short	input	C buffer height

height_c		Set greater than input picture vertical size/2.

6.11 T_FDP_REFBUF

The following is described about the member of T_FDP_REFBUF structure.

```
typedef struct{
        T_FDP_IMGBUF    *buf_refwr;
        T_FDP_IMGBUF    *buf_refrd0;
        T_FDP_IMGBUF    *buf_refrd1;
        T_FDP_IMGBUF    *buf_refrd2;
        T_FDP_IMGBUF    *buf_iirwr;
        T_FDP_IMGBUF    *buf_iirrd;
}T_FDP_REFBUF;
```

Table 6.11 member of T_FDP_REFBUF

Type/member	Input/output	Description
T_FDP_IMGBUF *buf_refwr	input	ignore
T_FDP_IMGBUF *buf_refrd0	input	Pointer to read buffer 0(next field) In following case, ignore this parameter(can set NULL) -Telecine mode(fproc_par->f_decodeseq=1) non-access pattern(*1) -First frame of 3D-IPC mode -Specify Fproc_par->last_start=1 in case of 3D-IPC mode -2D-IPC mode -Progressive mode(seq_mode=PROGRESSIVE) Other case not specify NULL
T_FDP_IMGBUF *buf_refrd1	input	Pointer to read buffer 1(current field) Do not specify NULL
T_FDP_IMGBUF *buf_refrd2	input	Pointer to read buffer 2(previous field) In following case, ignore this parameter(can set NULL) -Telecine mode(fproc_par->f_decodeseq=1) non-access pattern(*1) -First frame of 3D-IPC mode -Specify fproc_par->last_start=1 in case of 3D-IPC mode -2D-IPC mode -Progressive mode(seq_mode=PROGRESSIVE) Other cases not specify NULL.
T_FDP_IMGBUF *buf_iirwr	input	Unused this member
T_FDP_IMG_BUF *buf_iirrd	input	Unused this member

(*1)In figure 3-12 and 3-13, draw gray hatching in previous/next column.

6.12 T_FDP_STATUS

The following is described about the member of T_FDP_STATUS structure.

```
typedef struct{
        unsigned char       status;
        unsigned long       delay;
        unsigned long       vcycle;
        unsigned short      vintcnt;
        unsigned char       seq_lock;
        unsigned char       in_enable;
        unsigned long       in_picid;
        unsigned char       in_left;
        unsigned char       out_enable;
        unsigned long       out_picid;
        unsigned char       out_left;
        unsigned char       out_req;
}T_FDP_STATUS;
```

Table 6.12 member of T_FDP_STATUS

Type / member	Input/output	Describe
unsigned char status	output	Status of FDP manager FDPM_IDLE FDPM_RDY FDPM_SHARE_BUSY FDPM_OCCUPY_BUSY FDPM_FULL_BUSY
unsigned long delay	output	Status of delay 0:FDP processing finished in Vperiod Non-zero:FDP processing not finished in Vperiod. Indicate delay time(1=0.1ms) from Vperiod timimng. In Best Effort mode, always “0” indicate.
unsigned long vcycle	output	Number of FDP processing cycle
unsigned short vintcnt	output	Count value of V interrupts from drv_FDPM_Open. In Best Effort mode, always “0” indicate.
unsigned char seq_lock	output	Status of sequence lock FDP_SEQ_UNLOCK:unlock FDP_SEQ_LOCK: sequence lock(first field input status in interlace sequence)
unsigned char in_enable	output	Input picture enable FDP_IN_DISABLE:invalid picture FDP_IN_ENABLE:valid picture
unsigned long in_picid	Output	Input picture ID(if in_enable=”FDP_IN_ENABLE”, valid) Indidate immediate drv_FDPM_Start of fproc_par->in_pic->picid value.
unsigned in_left	output	Number of remain frame(if in_enable=”FDP_IN_ENABLE”, valid) Indicate remain frame of FDP processing for immediate drv_FDPM_Start. If “0” indicate, Set next input picture. In video sequence (fproc_par->f_decodeseq=0), always “0” indicate.
unsigned char out_enable	output	Output picture enable FDP_OUT_DISABLE: indicate no output picture FDP_OUT_ENABLE: indicate output picture

unsigned long out_picid	output	Output picture ID(if out_enable="FDP_OUT_ENABLE", valid)
unsigned long out_left	output	Number of remain output picture(if out_enable="FDP_OUT_ENBALE", valid)
unsigned char out_req	output	Output request FDP_OUT_NOREQ: no need to set output buffer when next drv_FDPM_Start call.(do not output) FDP_OUT_REQ: need to set output buffer when next drv_FDPM_Start call

7.Definition

7.1 Return Value Definition

table 7.1 table of return value definition

Definition	Value	Content
R_FDPM_OK	0	Success
R_FDPM_NG	-1	Error
EACCES	Depend on errno definition	Invalid access
EINVAL	Depend on errno definition	Invalid parameter
ENOMEM	Depend on errno definition	Not enough memory

7.2 Parameter Definition

table 7.2 table of parameter definition

Definition	Value	Content
TO_VCNT	1*10000 (=1sec)	Time out callback(callback4) period (1=0.1ms)

7.3 Sub error code definition

table 7.3 table of sub error code definition

Definition	Value	Content
E_FDP_PARA_CB3	-200	Invalid callback3 parameter
E_FDP_PARA_CB4	-201	Invalid callback4 parameter
E_FDP_PARA_OPENPAR	-202	Invalid open parameter
E_FDP_PARA_REFBUFMODE	-203	Invalid refbuf_Mode parameter
E_FDP_PARA_BUFREF0	-204	Invalid bufref0 parameter
E_FDP_PARA_BUFREF1	-205	Invalid bufref1 parameter
E_FDP_PARA_BUFREF2	-206	Invalid bufref2 parameter
E_FDP_PARA_BUFREFPRG	-207	Invalid bufrefprg parameter
E_FDP_PARA_VMODE	-208	Invalid vmode parameter

E_FDP_PARA_CLKMODE	-209	Invalid clkmode parameter
E_FDP_PARA_OCMODE	-210	Invalid ocmode parameter
E_FDP_PARA_INSIZE	-211	Invalid insize parameter
E_FDP_PARA_CB1	-250	Invalid callback1 parameter
E_FDP_PARA_CB2	-251	Invalid callback2 parameter
E_FDP_PARA_VCNT	-252	Invalid vcnt parameter
E_FDP_PARA_REFBUF	-253	Invalid refbuf parameter
E_FDP_PARA_BUFADDR	-254	Invalid buf_addr parameter
E_FDP_PARA_BUFADDRC	-255	Invalid buf_addrc parameter
E_FDP_PARA_BUFADDRC1	-256	Invalid buf_addrc1 parameter
E_FDP_PARA_BUFSTRIDE	-257	Invalid buf_stride parameter
E_FDP_PARA_BUFHEIGHT	-258	Invalid buf_height parameter
E_FDP_PARA_BUFHEIGHTC	-259	Invalid buf_heightc parameter
E_FDP_PARA_STARTPAR	-300	Invalid start_par parameter
E_FDP_PARA_FDPGO	-301	Invalid fdpgo parameter
E_FDP_PARA_FPROCPAR	-302	Invalid fproc_par parameter
E_FDP_PARA_SEQPAR	-303	Invalid seq_par parameter
E_FDP_PARA_IMGSETPAR	-304	Invalid imgset parameter
E_FDP_PARA_INPIC	-305	Invalid inpic parameter
E_FDP_PARA_OUTBUF	-306	Invalid outbuf parameter
E_FDP_PARA_SEQMODE	-307	Invalid seq_mode parameter
E_FDP_PARA_TELECINEMODE	-308	Invlid telecine_mode parameter
E_FDP_PARA_INWIDTH	-309	Invalid inwidth parameter
E_FDP_PARA_INHEIGHT	-310	Invalid inheight parameter
E_FDP_PARA_PICPAR	-311	Invalid picpar parameter
E_FDP_PARA_INBUF1	-312	Invalid inbuf1 parameter
E_FDP_PARA_INBUF2	-313	Invalid inbuf2 parameter
E_FDP_PARA_PICWIDTH	-314	Invalid pic_width parameter
E_FDP_PARA_PICHEIGHT	-315	Invalid pic_height parameter

E_FDP_PARA_CHROMA	-316	Invaplid chroma parameter
E_FDP_PARA_PROGSEQ	-317	Invalid progressive sequence parameter
E_FDP_PARA_PICSTRUCT	-318	Invaplid picture_structure parameter
E_FDP_PARA_REPEATTOP	-319	Invalid repeat_top_field parameter
E_FDP_PARA_BUFREFWR	-320	Invalid buf_refwr parameter
E_FDP_PARA_BUFREFRD0	-321	Invalid buf_refrd0 parameter
E_FDP_PARA_BUFREFRD1	-322	Invalid buf_refrd1 parameter
E_FDP_PARA_BUFREFRD2	-323	Invalid buf_refrd2 parameter
E_FDP_PARA_BUFIIRWR	-324	Invalid buf_iirwr parameter
E_FDP_PARA_BUFIIRRD	-325	Invalid buf_iirrd parameter
E_FDP_PARA_SEQOVERLAP	-326	Invalid sequence overlap parameter
E_FDP_PARA_FIELD_PARITY	-327	Invalid field parity parameter
E_FDP_PARA_STATUS	-328	Invalid status parameter
E_FDP_PARA_LASTSTART	-329	Invalid last_start parameter
E_FDP_PARA_CF	-330	Invalid cf parameter
E_FDP_PARA_FDECODE	-331	Invalid f_decode parameter
E_FDP_PARA_OUTFORMAT	-332	Invalid outformat parameter
E_FDP_CANCEL_NOID	-400	no cancel_id
E_FDP_CANCEL_ID_PROCESSING	-401	Cancel id’s frame is processing
E_FDP_TIMER_CB	-80	Callback timer error
E_FDP_TIMER_TO	-81	Time out timer error

8.Revision History

Revision History	FDP Manager for Linux(Ver.1.00) User's Manual: Software

Rev.	Date	Description	
		Page	Summary
1.00	May.2014	-	First edition issued
1.01	June.2014	9	Update Figure 1-1
		17	Add 2.1.5

FDP Manager for Linux User’s Manual: Software

Publication Date: Rev.1.01 June, 2014

Published by: Renesas Electronics Corporation

RENESAS

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

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

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

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

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

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

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

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

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

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

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

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

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

Colophon 1.3

FDP Manager for Linux
User's Manual: Software

Renesas Electronics Corporation