# OMAP35x EVM Linux PSP

# Migration Guide - Moving to Open source(GIT) OMAP35x Linux PSP

02.01.03.11

Publication date 10 September 2009

# IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

| **Products** | | **Applications** | |
|---|---|---|---|
| Amplifiers | amplifier.ti.com | Audio | www.ti.com/audio |
| Data Converters | dataconverter.ti.com | Automotive | www.ti.com/automotive |
| DSP | dsp.ti.com | Broadband | www.ti.com/broadband |
| Interface | interface.ti.com | Digital Control | www.ti.com/digitalcontrol |
| Logic | logic.ti.com | Military | www.ti.com/military |
| Power Mgmt | power.ti.com | Optical Networking | www.ti.com/opticalnetwork |
| Microcontrollers | microcontroller.ti.com | Security | www.ti.com/security |
| | | Security | www.ti.com/security |
| | | Telephony | www.ti.com/telephony |
| | | Video & Imaging | www.ti.com/video |
| | | Wireless | www.ti.com/wireless |

Mailing Address:

Texas Instruments,
Post Office Box 655303,
Dallas, Texas 75265

# Table of Contents

# List of Figures

# List of Tables

# Read This First

## About This Manual

This document describes interface level changes with respect to the 2.6.22 kernel based Linux PSP releases and should serve as an aid for migrating existing applications from PSP 1.0.x releases to the open source(GIT) based Linux PSP releases.

## How to Use This Manual

This document includes the following chapters:

- Chapter 1, *Fbdev Display Driver* - contains detailed information about Framebuffer driver related changes
- Chapter 2, *V4L2 Display Driver* - contains detailed information about V4L2 display driver related changes.
- Chapter 3, *V4L2 Capture Driver* - contains detailed information about V4L2 Capture driver related changes.

## Notation of information elements

The document may contain these additional elements:

**Warning**

This is an example of warning message. It usually indicates a non-recoverable change, e.g. formatting a filesystem.

**Caution**

This is an example of caution message.

**Important**

This is an example of important message.

**Note**

This is an example of additional note. This usually indicates additional information in the current context.

**Tip**

This is an example of a useful tip.

## If You Need Assistance

For any assistance, please send an mail to software support [mailto:softwaresupport@ti.com].

## Trademarks

OMAP™ is a trademark of Texas Instruments Incorporated.

All other trademarks are the property of the respective owner.

# Fbdev Display Driver

**Abstract**

## 1.1. Frame-buffer Driver

This document provides an detailed information of the User interface changes in Frame-buffer driver as compared with PSP1.0.x release, and plays the role of migration guide for the users from PSP1.0.x to GIT based release. The user interface changes are primarily because of the open source comments and new display library submitted to community. Please note that, all the user interfaces may not be part of GIT baseline and will be maintained as custom patches for the release purpose.

### 1.1.1. Features Supported

1. Supports LCD display interface at VGA (480x640) resolution.
2. Supports DVI display interface at 720P or 480P resolution through compile time option.

> **Note**
>
> Please note that work is in progress to add support for 720P and 480P, will try to include as a part of release.

3. Supports Switching of output through SYSFS interface.

4. Supports all the three planes (GFX, Vid1 and Vid2) via compile time option. The default configuration is GFX - FBDEV, Vid 1 - V4L2 and Vid2 - V4L2.
5. Supported color format: RGB888 and RGB565, RGB24(unpacked), ARGB, RGBA.
6. Configuration of parameters such as height and width of display screen, bits-per-pixel etc.
7. Supports rotation - 0, 90, 180 and 270 degrees.
8. Supports setting of global alpha value.
9. Panning is supported.
10. OMAPFB_WAIT_FOR_VSYNC ioctl is supported.
11. Supports backlight class driver, allowing user to control backlikght of LCD panel (Range is 0 -100).
12. Supports Power management (CPU Idle).

### 1.1.2. Features not supported compared to PSP1.0.2 release

1. Mirroring is not supported.
2. Alpha Blending is not supported.

### 1.1.3. List of IOCTLs same as PSP1.0.2 release

Following APIS/IOCTLs are not changed compare to PSP1.0.2 release. It can be used directly without any changes.

- FBIOGET_VSCREENINFO
- FBIOPUT_VSCREENINFO
- FBIOGET_FSCREENINFO
- FBIOGETCMAP
- FBIOPUTCMAP
- FBIO_BLANK
- FBIOPAN_DISPLAY

### 1.1.4. List of IOCTLs not supported

Following ioctls are not supported compared to PSP1.0.2 release. Failing to remove them from the application may result in compilation errors.

- FBIO_MIRROR

### 1.1.5. List of new IOCTLs in Git release

Following is the list of new IOCTLs supported in Git release compared to PSP1.0.2 release

- OMAPFB_QUERY_PLANE
- OMAPFB_SETUP_MEM
- OMAPFB_QUERY_MEM
- OMAPFB_GET_CAPS
- OMAPFB_SET_UPDATE_MODE
- OMAPFB_GET_UPDATE_MODE
- OMAPFB_WAIT_FOR_VSYNC

Please refer to the section 1.1.7 for detailed description on the above ioctls.

### 1.1.6. Usage of Changed ioctls in GIT release

#### 1.1.6.1. Command Line arguments and Module params

FBDEV driver supports command line argument or module parameter for enabling/setting rotation angle. User has to enable rotation through command line option first to be able to use rotation through either SYSFS of through ioctl.

**PSP1.0.x Method**

```
OMAP3EVM# setenv bootargs console=ttyS0,115200n8 mem=128M
 noinitrd root=/dev/nfs nfsroot=172.24.133.229:/home/user/
remote/_install, nolock, size=1024, wsize=1024 ip=dhcp
 video=omap24xxfb:rotation=90 mpurate=600
```

**GIT Method**

```
 OMAP3EVM# setenv bootargs console=ttyS0,115200n8 mem=128M
 noinitrd root=/dev/nfs nfsroot=172.24.133.229:/home/user/remote/
_install, nolock, size=1024, wsize=1024 ip=dhcp omapfb.rotate=1
 omapfb.rotate_type=1 mpurate=600

where,
    rotate => 0 - 0 degree, 1 - 90 degree, 2 - 180 degree and 3 -
 270 degree respectively.
    rotate_type => 0 - DMA based rotation, 1 - VRFB based
 rotation. Currently only VRFB based rotation is supported.
```

**Note**

The entire command should be entered in a single line.

#### 1.1.6.2. Back-light control

User can control the backlight through SYSFS interface -
**PSP1.0.x Method**

```
SYSFS Attribute - /sys/class/display_control/omap_disp_control/
lcdbacklight

          Parameter -
            on - To turn on the LCD backlight
            Off - To turn off the LCD backlight
```

**GIT Method**

```
          SYSFS Attribute - /sys/class/backlight/omap3evm-
bklight/
            - actual_brightness
            - bl_power
            - brightness
            - max_brightness

          Parameter -
            0 - 100 - User can configure the LCD brightness
 between 0 - 100.
```

#### 1.1.6.3. Rotation

In both PSP1.0.x and GIT, FBDEV driver supports rotation by using rotation engine in the VRFB module. Rotation engine supports rotation of an image with degree 0, 90, 180 and 270. For using this feature of the driver, rotation has to be enabled. Application can enable rotation by enabling/setting rotation angle in boot time argument of the kernel for FBDEV driver. Please refer to the section 1.1.6.1. Currently, rotation is only supported for 16 bits-per-pixel display mode.
**PSP1.0.x Method** To provide an API for rotation, one of the 'reserved' fields in the `fb_var_screeninfo` structure is changed to `rotate` field. Applications can thus use the `FBIOPUT_VSCREENINFO` ioctl to set the rotation angle. Applications have to set this `rotate` field in the `fb_var_screeninfo` structure equal to the angle of rotation (0, 90, 180 or 270) and call this ioctl.

```
struct fb_var_screeninfo var;
/* Get the Variable screen info through "FBIOGET_VSCREENINFO */
```

```
var.rotate = 90; /* To set rotation angle to 90 degree */
if (ioctl(fb, FBIOPUT_VSCREENINFO, &var)<0) {
 perror("Error:FBIOPUT_VSCREENINFO\n");
 exit(4);
}
```

**GIT Method**

The rotation has been supported either through SYSFS entry or ioctl

SYSFS Interface

```
SYSFS Attribute - /sys/class/graphics/fb0/rotate

          0 - 0 degree, 1 - 90 degree, 2 - 180 degree and 3 - 270
 degree respectively
```

IOCTL interface

```
struct fb_var_screeninfo var;
/* Get the Variable screen info through "FBIOGET_VSCREENINFO" */
var.rotate = 1; /* To set rotation angle to 90 degree */
if (ioctl(fb, FBIOPUT_VSCREENINFO, &var)<0) {
 perror("Error:FBIOPUT_VSCREENINFO\n");
 exit(4);
}
```

#### 1.1.6.4. Buffer Management

| Driver | Without Rotation | With Rotation |
|---|---|---|
| FBDEV Driver (PSP1.0.x) | A single buffer of size 1280*720*4*2 bytes | A single buffer of size 2048*720*4 bytes |
| FBDEV Driver (GIT2.01.00.01) | A single buffer of size 1280*720*4*2 bytes | A single buffer of size 2048*720*4 bytes |

**Table 1.1. Memory requirement for FBDEV driver Buffers**

If rotation is enabled, driver allocates single buffer of maximum resolution for the VRFB memory space (2048x720x4).

#### 1.1.6.5. Buffer Format

TBD

#### 1.1.6.6. Color Lookup Table

The graphics pipeline supports the color look up table. The CLUT mode uses the encoded pixel values from the input image as pointers to index the 24-bit-wide CLUT value. `FBIOPUTCMAP` and `FBIOGETCMAP` can be used to set and get the color map table. When CLUT is set, the driver

makes the hardware to reload the CLUT. Following example shows how to change CLUT

```
struct fb_cmap cmap;
unsigned short r[4]={0xFF,0x00, 0x00, 0xFF};
unsigned short g[4]={0x00, 0xFF, 0x00, 0xFF};
unsigned short b[4]={0x00, 0x00, 0xFF, 0x00};
cmap.len = 4;
cmap.red = r;
cmap.green = g;
cmap.blue = b;
if (ioctl(fd, FBIOPUTCMAP, &cmap)) {
 perror("FBIOPUTCMAP\n");
 exit(3);
}
```

#### 1.1.6.7. Setting the global alpha value for graphics pipeline

PSP1.0.x was supporting the global alpha value through sysfs entries. PSP02.0x.0x supports the global alpha value through standard V4L2 ioctl.

**PSP1.0.x Method**

```
echo 255 > /sys/class/display_control/omap_disp_control/
gfx_global_alpha
 echo 0 > /sys/class/display_control/omap_disp_control/
gfx_global_alpha
```

**GIT Method**

```
struct fb_var_screeninfo var;
int global_alpha = 128;

if (ioctl(fb, FBIOGET_VSCREENINFO, &var)) {
 perror("Error reading variable information.\n");
 exit(3);
}
var.reserved[0] = global_alpha;
if (ioctl(fb, FBIOPUT_VSCREENINFO, &var)) {
 perror("Error writing variable information.\n");
 exit(3);
}
```

**Note**

Before using the global alpha or pixel based alpha on graphics pipeline, Alpha blending needs to be enabled using V4L2 ioctl. Please refer to the V4L2 display driver section "Enabling/Disabling the alpha blending" for further details.

#### 1.1.6.8. Panning the Display

The PSP1.0.x release was supporting panning of display without any constraint on user.

While PSP02.0x.0x release has buffer limitation, it expects user to allocate sufficient buffer size through `bootargs`.

**PSP1.0.x Method**

Follows standard FBDEV interface.
**GIT Method**

Git release also supports standard interface for panning, provided that user is not supposed to override size of buffer through bootargs as shown below OR allocate sufficient amount of memory.

```
 OMAP3EVM# setenv bootargs console=ttyS0,115200n8 mem=128M
 noinitrd root=/dev/nfs nfsroot=172.24.133.229:/home/user/
remote/_install, nolock, size=1024, wsize=1024 ip=dhcp
 omapfb.vram=7372800 mpurate=600

Where,
    vram = 1280 * 720 * 4 * 2
```

**Note**

Please note that panning is not supported when rotation is enabled in both PSP1.0.x and Git (02.0x.0x) release.

#### 1.1.6.9. Wait For VSYNC

The PSP1.0.x release was supporting ioctl `FBIO_WAITFORVSYNC`, which will allow user application to wait till VSYNC interrupt arrives.

While in the GIT release, the functionality is being supported through similar ioctl `OMAPFB_WAIT_FOR_VSYNC`.

Following example shows the usecase from user application point of view for -
**PSP1.0.x Method**

```
/* Wait for the current frame buffer to get displayed. */
ret = ioctl(fd, FBIO_WAITFORVSYNC, 0);
if(ret < 0) {
        perror("FBIO_WAITFORVSYNC\n");
        close(fd);
        exit(0);
}
```

**GIT Method**

```
/* Wait for the current frame buffer to get displayed. */
ret = ioctl(fd, OMAPFB_WAIT_FOR_VSYNC, 0);
if(ret < 0) {
        perror("OMAPFB_WAIT_FOR_VSYNC\n");
        close(fd);
        exit(0);
}
```

## 1.1.7. New Custom IOCTL's

### 1.1.7.1. OMAPFB_SYNC_GFX

Not being Tested.

### 1.1.7.2. OMAPFB_UPDATE_WINDOW

Not being Tested.

### 1.1.7.3. OMAPFB_SETUP_PLANE

Not being Tested.

### 1.1.7.4. OMAPFB_QUERY_PLANE

Query the plane (gfx) and returns the omapfb_plane_info information -

Data Structure

```
struct omapfb_plane_info {
__u32 pos_x;
__u32 pos_y;
__u8  enabled;
__u8  channel_out;
__u8  mirror;
__u8  reserved1;
__u32 out_width;
__u32 out_height;
__u32 reserved2[12];
            };
```

Usage:

```
struct omapfb_plane_info pi;
if (ioctl(fb, OMAPFB_QUERY_PLANE, &pi)) {
 perror("Error: OMAPFB_QUERY_PLANE.\n");
 exit(1);
```

```
 }
```

#### 1.1.7.5. OMAPFB_SETUP_MEM

Allows user to setup the frame buffer memory, like size and type.

Data Structure

```
#define OMAPFB_MEMTYPE_SDRAM            0
#define OMAPFB_MEMTYPE_SRAM             1
#define OMAPFB_MEMTYPE_MAX              1

struct omapfb_mem_info {
 __u32 size;
 __u8  type;
 __u8  reserved[3];
};
```

Usage:

```
struct omapfb_mem_info mi;
mi.size = > Expected size of buffer <
mi.type = > Expected type of buffer <
if (ioctl(fb, OMAPFB_SETUP_MEM, &mi)) {
 perror("Error: OMAPFB_QUERY_MEM.\n");
 exit(1);
}
```

**Note**

Please note that unexpected behaviour may occur after setting the buffer size to wrong value.

#### 1.1.7.6. OMAPFB_QUERY_MEM

Returns the size and type of the frame buffer.

Data Structure

```
#define OMAPFB_MEMTYPE_SDRAM            0
#define OMAPFB_MEMTYPE_SRAM             1
#define OMAPFB_MEMTYPE_MAX              1

struct omapfb_mem_info {
 __u32 size;
 __u8  type;
 __u8  reserved[3];
 };
```

Usage:

```
struct omapfb_mem_info mi;
if (ioctl(fb, OMAPFB_QUERY_MEM, &mi)) {
 perror("Error: OMAPFB_QUERY_MEM.\n");
 exit(1);
}
printf("size - %d\n", mi.size);
printf("type - %d\n", mi.type);
```

#### 1.1.7.7. OMAPFB_GET_CAPS

Not being Tested.

#### 1.1.7.8. OMAPFB_SET_UPDATE_MODE

Sets the current update mode.

```
enum omap_dss_update_mode {
OMAP_DSS_UPDATE_DISABLED = 0,
OMAP_DSS_UPDATE_AUTO,
OMAP_DSS_UPDATE_MANUAL,
};
```

```
OMAP_DSS_UPDATE_MANUAL is not supported currently.
OMAP_DSS_UPDATE_AUTO - Enables the DSS.
OMAP_DSS_UPDATE_DISABLED - Disables the DSS.
```

Usage:

```
enum omapfb_update_mode mode;
mode = OMAPFB_UPDATE_DISABLED;
if (ioctl(fb, OMAPFB_SET_UPDATE_MODE, &mode)) {
 perror("Error: OMAPFB_GET_UPDATE_MODE.\n");
 exit(1);
}
```

**Note**

Please note that this IOCTL is not supported for TV out

#### 1.1.7.9. OMAPFB_GET_UPDATE_MODE

This returns the current update mode of DSS

```
 enum omap_dss_update_mode {
```

```
 OMAP_DSS_UPDATE_DISABLED = 0,
 OMAP_DSS_UPDATE_AUTO,
 OMAP_DSS_UPDATE_MANUAL,
};

OMAP_DSS_UPDATE_MANUAL is not supported currently.
OMAP_DSS_UPDATE_AUTO - indicates the DSS is enabled.
OMAP_DSS_UPDATE_DISABLED - indicates DSS is disabled.
```

Usage:

```
enum omapfb_update_mode mode;
if (ioctl(fb, OMAPFB_GET_UPDATE_MODE, &mode)) {
 perror("Error: OMAPFB_GET_UPDATE_MODE.\n");
 exit(1);
}
printf("mode - %d\n", mode);
```

**Note**

Please note that this IOCTL is not supported for TV out

#### 1.1.7.10. OMAPFB_LCD_TEST

Not being Tested.

#### 1.1.7.11. OMAPFB_CTRL_TEST

Not being Tested.

### 1.1.8. Supported Module Params

#### 1.1.8.1. DSS Library supported Params

##### 1.1.8.1.1. Default Display

User can configure the default display output during boot-time using bootargs, as shown below

```
 OMAP3EVM# setenv bootargs console=ttyS0,115200n8 mem=128M
 noinitrd root=/dev/nfs nfsroot=172.24.133.229:/home/user/
remote/_install, nolock, size=1024, wsize=1024 ip=dhcp omap-
dss.def_disp="dvi" mpurate=600

       Where,
         dvi = Default output goes to DVI display.
         tv = Default output goes to TV display.

  Without any bootarg configuration, the deffault output goes to
  LCD.
```

**Note**

The entire command should be entered in a single line.

#### 1.1.8.2. Frame-Buffer Driver Supported Params

##### 1.1.8.2.1. Default FrameBuffer size

User can configure the maximum size of the framebuffer through boot-agrs, as shown below -

```
 OMAP3EVM# setenv bootargs console=ttyS0,115200n8 mem=128M
 noinitrd root=/dev/nfs nfsroot=172.24.133.229:/home/user/
remote/_install, nolock, size=1024, wsize=1024 ip=dhcp vram=10M
 omapfb.vram=4M,3M,3M mpurate=600

       Where,
         vram=10M : Maximum size of buffer which system can use
 for FB.
         omapfb.vram=4M,3M,3M : This will allocate 4M for
 fb0(GFX), 3M
  for fb1(Vid1) and 3M for fb2(vid2).

  Without any bootarg configuration, only buffer for fb0 will be
  allocated.
```

**Note**

The entire command should be entered in a single line.

**Note**

Please note that, configuring size of buffer through bootargs will override all the driver settings. So please ensure that you allocate suffcient amount of memory, else behaviour is unpredictable

##### 1.1.8.2.2. Video Mode Params

User can configure the default mode of configuration and corrosponding timing parameters through bootargs, as shown below -

```
 OMAP3EVM# setenv bootargs console=ttyS0,115200n8 mem=128M
 noinitrd root=/dev/nfs nfsroot=172.24.133.229:/home/user/
remote/_install, nolock, size=1024, wsize=1024 ip=dhcp
 omapfb.video_mode=720x576@50 mpurate=600

       Where,
```

```
        720x576@50 : width x height @ Hz.

      Please make sure that the mode setting is being supported
by FB.
      To add new mode setting to FB please edit modedb.c file
with
      appropiate parameters.
```

**Note**

The entire command should be entered in a single line.

#### 1.1.8.2.3. VRFB Rotation related params

Please refer to the section 1.1.6.1.

### 1.1.9. SYSFS Software Design Interface

#### 1.1.9.1. Frame-buffer Driver: Standard sysfs attributes

Following attributes are available for user control -

```
[root@OMAP3EVM /]# ls -l /sys/class/graphics/fb0/

-rw-r--r--  1 root root  4096 Jan  1 00:00 bits_per_pixel
-rw-r--r--  1 root root  4096 Jan  1 00:00 blank
-rw-r--r--  1 root root  4096 Jan  1 00:00 console
-rw-r--r--  1 root root  4096 Jan  1 00:00 cursor
-r--r--r--  1 root root  4096 Jan  1 00:00 dev
lrwxrwxrwx  1 root root     0 Jan  1 00:00 device -> ../../../
devices/platform/omapfb
-rw-r--r--  1 root root  4096 Jan  1 00:00 mode
-rw-r--r--  1 root root  4096 Jan  1 00:00 modes
-r--r--r--  1 root root  4096 Jan  1 00:00 name
-rw-r--r--  1 root root  4096 Jan  1 00:00 pan
drwxr-xr-x  2 root root     0 Jan  1 00:00 power
-rw-r--r--  1 root root  4096 Jan  1 00:00 rotate
-rw-r--r--  1 root root  4096 Jan  1 00:00 rotate_type
-rw-r--r--  1 root root  4096 Jan  1 00:00 state
-r--r--r--  1 root root  4096 Jan  1 00:00 stride
lrwxrwxrwx  1 root root     0 Jan  1 00:00 subsystem -> ../../
graphics
-rw-r--r--  1 root root  4096 Jan  1 00:00 uevent
-rw-r--r--  1 root root  4096 Jan  1 00:00 virtual_size
```

**Note**

Please note that the above attributes are same as PSP1.0.x, but the behavior/functionality is different. This section only covers GIT part of it

| SYSFS attribute | Description |
|---|---|
| bits_per_pixel | Allows user to control bits per pixel configuration, currently the supported values are 16 and 24.<br><br>`# echo 16/24 > /sys/class/graphics/fb0/bits_per_pixel` |
| blank | Allows user to control lcd display blanking configuration independently.<br><br>`echo 0/4  > /sys/class/graphics/fb0/blank`<br><br>`Values only 0(FB_BLANK_UNBLANK) and 4(FB_BLANK_POWERDOWN) is supported.` |
| mode | TBD. |
| modes | TBD |
| rotate | Allows user to control rotation through this entry,<br><br>`# echo 0/1/2/3 > /sys/class/graphics/fb0/rotate`<br><br>`0 - 0 degree, 1 - 90 degree, 2 - 180 degree and 3 - 270 degree respectively.` |
| rotate_type | Allows user to control rotation type through this entry,<br><br>`# echo 0/1 > /sys/class/graphics/fb0/rotate_type`<br><br>`0 - DMA based rotation,`<br>`1 - VRFB based rotation.`<br>`Currently only VRFB based rotation is supported.` |
| virtual_size | Allows user to configure xres_virtual and yres_virtual parameters of frame-buffer,<br><br>`TBD` |

**Table 1.2. Frame-buffer Driver: Standard sysfs attributes**

#### 1.1.9.2. DSS Library: sysfs attributes

Following attributes are supported -

```
[root@OMAP3EVM]# ls -l /sys/devices/platform/omap-dss/

lrwxrwxrwx 1 root root    0 Jan  1 01:46 bus -> ../../../bus/
platform
-r--r--r-- 1 root root 4096 Jan  1 01:46 clk
lrwxrwxrwx 1 root root    0 Jan  1 01:46 driver -> ../../../bus/
platform/drivers/omap-dss
-r--r--r-- 1 root root 4096 Jan  1 01:46 modalias
drwxr-xr-x 2 root root    0 Jan  1 01:46 power
lrwxrwxrwx 1 root root    0 Jan  1 01:46 subsystem -> ../../../
bus/platform
-rw-r--r-- 1 root root 4096 Jan  1 01:46 uevent
[root@OMAP3EVM]#
```

| SYSFS attribute | Description |
|---|---|
| clk | This is read only entry, which will display the current clock settings for DSS. |

```
[root@OMAP3EVM]# cat /sys/devices/platform/
omap-dss/clk
- dss -
internal clk count           2
dss_ick         83000000     1
dss1_alwon_fck  96000000     1
dss2_alwon_fck  1_tv_fck     54000000  0
dss_96m_fck     96000000     0
- dispc -
dispc fclk source = dss1_alwon_fclk
pixel clk = 96000000 / 1 / 5 = 19200000
[root@OMAP3EVM]#
```

**Table 1.3. DSS Library: sysfs attributes**

#### 1.1.9.3. Frame-buffer Driver: Custom sysfs attributes

Following attributes are available for user control -

```
[root@OMAP3EVM]# ls -l /sys/devices/platform/omapfb/

lrwxrwxrwx 1 root root     0 Jan  1 01:54 bus -> ../../../bus/
platform
-rw-r--r-- 1 root root  4096 Jan  1 01:54 displays
lrwxrwxrwx 1 root root     0 Jan  1 01:54 driver -> ../../../bus/
platform/drivers/omapfb
-rw-r--r-- 1 root root  4096 Jan  1 01:54 frame-buffers
lrwxrwxrwx 1 root root     0 Jan  1 01:54 graphics:fb0 -
> ../../../class/graphics/fb0
-rw-r--r-- 1 root root  4096 Jan  1 01:54 managers
-r--r--r-- 1 root root  4096 Jan  1 01:54 modalias
-rw-r--r-- 1 root root  4096 Jan  1 01:54 overlays
drwxr-xr-x 2 root root     0 Jan  1 01:54 power
-rw-r--r-- 1 root root  4096 Mar 11 23:57 sleep_timeout
```

```
lrwxrwxrwx 1 root root     0 Jan  1 01:54 subsystem -> ../../../
bus/platform
-rw-r--r-- 1 root root  4096 Jan  1 01:54 uevent
[root@OMAP3EVM]#
```

| SYSFS attribute | Description |
|---|---|
| displays | Lists all the available supported displays with the following format -<br><br>`<display name>`<br>`e:<enabled>`<br>`u:<update mode>`<br>`t:<tear sync on/off>`<br>`h:<xres/hfp/hbp/hsw>`<br>`v:<yres/vfp/vbp/vsw>`<br>`p:<pix clock, in kHz>`<br>`m:<mode str, as in drivers/video/modedb.c:fb_find_mode>`<br><br>`[root@OMAP3EVM]# cat /sys/devices/platform/omapfb/displays`<br><br>`lcd e:1 u:1 t:0 h:480/1/28/2 v:640/1/1/1 p:19200 r:0 i:0`<br>`dvi e:0 u:1 t:0 h:720/48/80/32 v:480/3/7/4 p:30000 r:0 i:0`<br>`tv e:0 u:-1 t:0 h:720/0/0/0 v:574/0/0/0 p:0 r:0 i:0`<br>`[root@OMAP3EVM]#` |
| framebuffers | Lists all the available framebuffers with the following format -<br><br>`<fb number>`<br>`p:<physical address, read only>`<br>`v:<virtual address, read only>`<br>`s:<size, read only>`<br>`t:<target overlay>`<br><br>`[root@OMAP3EVM]# cat /sys/devices/platform/omapfb/framebuffers`<br><br>`0 p:87300000 v:ff245000 size:614400 t:gfx`<br>`[root@OMAP3EVM]#` |

| SYSFS attribute | Description |
|---|---|
| managers | Lists all the overlay managers with the following format - |

| SYSFS attribute | Description |
|---|---|
| | `<manager name>`<br>`t:<target display>`<br><br>`[root@OMAP3EVM]# cat /sys/devices/platform/omapfb/managers`<br><br>`lcd t:lcd`<br>`tv t:tv`<br>`l4 t:none`<br>`[root@OMAP3EVM]#` |
| overlays | Lists all the overlays with the following format -<br><br>`<overlay name>`<br>`t:<target manager>`<br>`x:<xpos>`<br>`y:<ypos>`<br>`iw:<input width, read only>`<br>`ih:<input height, read only>`<br>`w:<output width>`<br>`h:<output height>`<br>`e:<enabled>`<br><br>`[root@OMAP3EVM]# cat /sys/devices/platform/omapfb/overlays`<br><br>`gfx t:lcd x:0 y:0 iw:480 ih:640 w:480 h:640 e:1`<br>`vid1 t:lcd x:0 y:0 iw:160 ih:120 w:160 h:120 e:0`<br>`vid2 t:lcd x:0 y:0 iw:160 ih:120 w:160 h:120 e:0`<br>`l4-ovl t:l4 x:0 y:0 iw:0 ih:0 w:0 h:0 e:0`<br>`[root@OMAP3EVM]#` |

**Table 1.4. Frame-buffer Driver: Custom sysfs attributes**

| SYSFS attribute | Description |
|---|---|
| sleep_timeout | Sleep time-out feature for frame-buffer driver. This is global time-out feature for all fb nodes (fb0, fb1, fb2,...). The default timeout value is 20 Sec.<br><br>`# cat /sys/devices/platform/omapfb/sleep_timeout`<br>`20` |

| SYSFS attribute | Description |
|---|---|
| | To over-write default timeout value<br><br>`# echo 100 > /sys/devices/platform/omapfb/sleep_timeout` |

**Note**

Please note that, with current FileSystem the FBDEV keeps on refreshing the timer value after timeout, due to demo application.

**Table 1.5. Frame-buffer Driver: Custom sysfs attributes**

#### 1.1.9.4. Switching of output

This section explains and illustrates how user can switch the output dynamically and corresponding constraints wherever applicable.

##### 1.1.9.4.1. Switching of output device from LCD to DVI

Issue following commands to switch output device from LCD to DVI

```
# echo "gfx e:0" > /sys/devices/platform/omapfb/overlays
# echo "lcd e:0" > /sys/devices/platform/omapfb/displays
# echo "lcd t:none" > /sys/devices/platform/omapfb/managers
# echo "dvi e:1" > /sys/devices/platform/omapfb/displays
# echo "lcd t:dvi" > /sys/devices/platform/omapfb/managers
# Set the Frame buffer parameters like xres, xres_virtual, y_res,
 yres_virtual
again before running below command. For example for 720P as DVI
 resolution set
xres=1280 yres=720 xres_virtual=1280 yres_virtual=1440.
# echo "gfx e:1" > /sys/devices/platform/omapfb/overlays
```

##### 1.1.9.4.2. Switching of output device from DVI to LCD

Issue following commands to switch output device from DVI to LCD

```
# echo "gfx e:0" > /sys/devices/platform/omapfb/overlays
# echo "dvi e:0" > /sys/devices/platform/omapfb/displays
# echo "lcd t:none" > /sys/devices/platform/omapfb/managers
# echo "lcd e:1" > /sys/devices/platform/omapfb/displays
# echo "lcd t:lcd" > /sys/devices/platform/omapfb/managers
# Set the Frame buffer parameters like xres, xres_virtual, y_res,
 yres_virtual
again before running below command. For example for 640X480 set
xres=480 yres=640 xres_virtual=480 yres_virtual=1280
# echo "gfx e:1" > /sys/devices/platform/omapfb/overlays
```

#### 1.1.9.4.3. Switching GFX plane from LCD to TV

Issue following commands to switch GFX plane output from LCD to TV -

```
# echo "gfx e:0" > /sys/devices/platform/omapfb/overlays
# echo "gfx t:tv e:0" > /sys/devices/platform/omapfb/overlays
# echo "tv e:1" > /sys/devices/platform/omapfb/displays
# Set the Frame buffer parameters like xres, xres_virtual, y_res,
 yres_virtual
again before running below command. For example for NSTC_M set
xres=720 yres=480 xres_virtual=720 yres_virtual=960
# echo "gfx e:1" > /sys/devices/platform/omapfb/overlays
```

#### 1.1.9.4.4. Switching GFX plane from TV to LCD

Issue following commands to switch GFX plane output from LCD to TV -

```
# echo "gfx e:0" > /sys/devices/platform/omapfb/overlays
# echo "gfx t:lcd e:0" > /sys/devices/platform/omapfb/overlays
# echo "lcd e:1" > /sys/devices/platform/omapfb/displays
# Set the Frame buffer parameters like xres, xres_virtual, y_res,
 yres_virtual
again before running below command. For example for 640X480 set
xres=480 yres=640 xres_virtual=480 yres_virtual=1280
# echo "gfx e:1" > /sys/devices/platform/omapfb/overlays
```

#### 1.1.9.4.5. Switching Vid1 Plane from LCD to TV

Issue following commands to switch Vid1 plane output from LCD to TV -

```
# echo "vid1 e:0" > /sys/devices/platform/omapfb/overlays
# echo "vid1 t:tv e:0" > /sys/devices/platform/omapfb/overlays
# echo "tv e:1" > /sys/devices/platform/omapfb/displays
```

#### 1.1.9.4.6. Switching Vid1 Plane from TV to LCD

Issue following commands to switch Vid2 plane output from TV to LCD -

```
# echo "vid2 e:0" > /sys/devices/platform/omapfb/overlays
# echo "vid2 t:lcd e:0" > /sys/devices/platform/omapfb/overlays
# echo "lcd e:1" > /sys/devices/platform/omapfb/displays
```

#### 1.1.9.4.7. Switching Vid2 Plane from LCD to TV

Issue following commands to switch Vid2 plane output from LCD to TV -

```
# echo "vid2 e:0" > /sys/devices/platform/omapfb/overlays
# echo "vid2 t:tv e:0" > /sys/devices/platform/omapfb/overlays
```

```
# echo "tv e:1" > /sys/devices/platform/omapfb/displays
```

#### 1.1.9.4.8. Switching Vid2 Plane from TV to LCD

Issue following commands to switch Vid2 plane output from TV to LCD -

```
# echo "vid2 e:0" > /sys/devices/platform/omapfb/overlays
# echo "vid2 t:lcd e:0" > /sys/devices/platform/omapfb/overlays
# echo "lcd e:1" > /sys/devices/platform/omapfb/displays
```

**Note**

Streaming should be disabled on Video pipelines which changing the ouputs and overlay managers on video pipelines

**Note**

Outputs cannot be switched to currently active output. If the active ouput is LCD on LCD overlay manager then enabling the LCD again will make sysfs entries unstable.

Currently switching of output from the frame buffer timer and sysfs entries is not protected. so when the streaming is ON on the V4L2 driver or application is switching outputs till that time the timer should not be triggered. Following is the command to increase the time out value.
`echo <timeout value in seconds> > /sys/devices/platform/omapfb/sleep_timeout`

If the timer is disabled then system may not go into the offmode even if the system is idle

**Constraints/Known issues**

1. Enabling and disabling the tv display may run into problem due to the known issue with VENC. So it is recommended that user should enable the tv display once and switch the output to and from LCD and TV using command. *echo "tv e:1" > /sys/devices/platform/omapfb/displays*

### 1.1.9.5. Configuration of display Modes

Git version of release doesn't support dynamic configuration of modes, unlike PSP1.0.x release. User can change the modes through compile time option provided.

#### 1.1.9.5.1. Configuring DVI modes

To Select 480P or 720P:

Open menuconfig options from kernel command prompt. Go to the Device Drivers => Graphics support => OMAP2/3 Display Device Drivers, enable Generic Panel and choose default mode as either 480P or 720P as shown below.

Select Device Drivers as shown here:

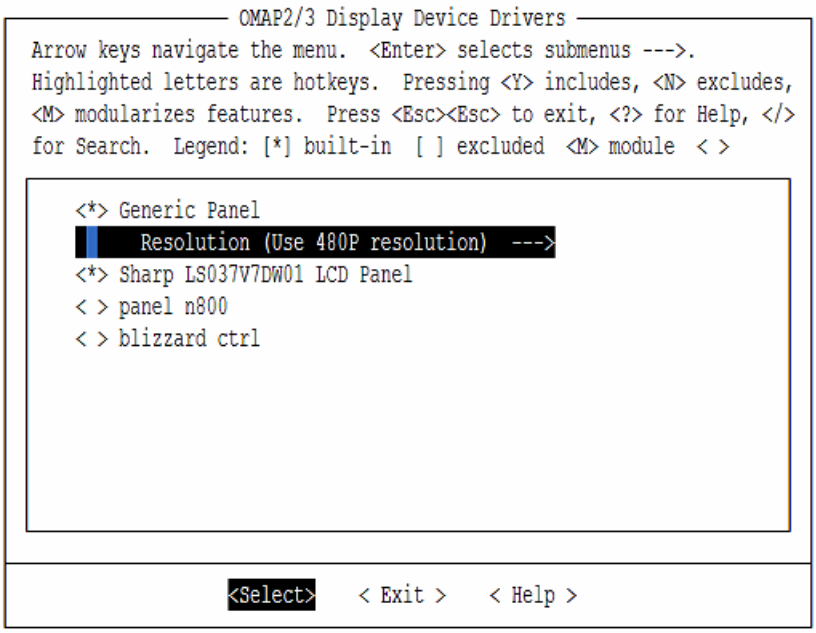


**Figure 1.1. Configure DVI Modes (720P/480P): Step 2**

**Note**

Please refer to the UserGuide to enable new frame buffer driver based on DSS2 library.

# V4L2 Display Driver

**Abstract**

## 2.1. V4L2 display Driver

This document provides the detailed usage of the user interface API changes in the V4L2 driver of GIT release of PSP compared to PSP1.0.2 release. This user interface changes are primarily because of the open source comments and new display library submitted to community.

### 2.1.1. Features Supported

1. V4L2 interface controls the two of video pipelines of the DSS.
2. V4L2 display driver supports TV display interface at NTSC and PAL resolutions on Video Pipelines (only S-Video out is supported, composite out is not supported)
3. Configuration of parameters such as height and width of display screen, pixel format etc.
4. V4L2 display driver supports buffer management through memory mapped and user pointer buffer exchange for application usage (mmaped).
5. Streaming support on both video pipelines.
6. V4L2 display driver supports rotation - 0, 90, 180 and 270 degrees on TV and LCD output.

7. Scaling supported from 8X to 1/2x. Scaling coefficients are not updated as per PSP1.0.2 release. Scaling coefficients are from the open source DSS library

8. Cropping and Windowing supported.

9. Configuration of parameters such as height and width of display screen, bits-per-pixel etc.

10. Alpha blending supported. Both pixel alpha and global alpha.

11. Background color setting supported.

12. Supports destination and source colorkeying on Video pipelines through V4L2 ioctls.

### 2.1.2. Features not supported compared to PSP1.0.2 release

1. Linking not supported.

2. No IOCTL to change the color space conversion coefficients from YUV to RGB. Default coefficients for color space conversion programmed.

3. No Power management supports.

4. No Off mode supports.

5. Down scaling not supported beyong 1/2x.

6. Change of Modes on TV not supported. Default mode is PAL.

### 2.1.3. List of IOCTLs same as PSP1.0.2 release

Following APIS/IOCTLs are not changed compare to PSP1.0.2 release. It can be used directly without any changes.

Following is the list of IOCTLs supported in GIT release which are same as PSP1.0.2 release.

- VIDIOC_QUERYBUF
- VIDIOC_REQBUFS
- VIDIOC_QBUF
- VIDIOC_DQBUF
- VIDIOC_STREAMON
- VIDIOC_STREAMOFF
- VIDIOC_S_FMT

- VIDIOC_G_FMT
- VIDIOC_ENUM_FMT
- VIDIOC_TRY_FMT
- VIDIOC_S_CROP
- VIDIOC_G_CROP
- VIDIOC_QUERYCAP
- VIDIOC_CROPCAP

### 2.1.4. List of IOCTLs not supported

Following ioctls are not supported compared to PSP1.0.2 release. Failing to remove them from the application may result in compilation errors.

- VIDIOC_S_OMAP2_MIRROR
- VIDIOC_G_OMAP2_MIRROR
- VIDIOC_S_OMAP2_COLORKEY
- VIDIOC_G_OMAP2_COLORKEY
- VIDIOC_S_OMAP2_BGCOLOR
- VIDIOC_G_OMAP2_BGCOLOR
- VIDIOC_OMAP2_COLORKEY_ENABLE
- VIDIOC_OMAP2_COLORKEY_DISABLE
- VIDIOC_S_OMAP2_COLORCONV
- VIDIOC_G_OMAP2_COLORCONV
- VIDIOC_S_OMAP2_DEFCOLORCONV
- VIDIOC_OMAP2_COLORKEY_ENABLE
- VIDIOC_S_OMAP2_COLORKEY
- VIDIOC_S_OMAP2_LINK
- VIDIOC_G_OMAP2_LINK
- VIDIOC_G_OUTPUT
- VIDIOC_S_OUTPUT
- VIDIOC_ENUMOUTPUT

### 2.1.5. List of new IOCTLs in Git release

Following is the list of new IOCTLs supported in Git release compared to PSP1.0.2 release

- VIDIOC_G_CTRL
- VIDIOC_S_CTRL
- VIDIOC_QUERYCTRL
- VIDIOC_S_FBUF
- VIDIOC_G_FBUF

### 2.1.6. Usage of New/Changed ioctls in GIT release.

Following section describes the usage of the new/changed ioctls in the Git version of the V4L2 display driver compared to PSP1.0.2 release.

#### 2.1.6.1. Setting Rotation

PSP1.0.2 release is supporting the rotation through custom ioctls. While git release will support the rotation through the control id. Patch is submitted on to community to add the control id for rotation.

**PSP1.0.x Method**

```
int rotation = 90;
ret = ioctl(display_fd, VIDIOC_S_OMAP2_ROTATION, &rotation);
if(ret < 0) {
    perror("VIDIOC_S_OMAP2_ROTATION\n");
    close(display_fd);
    exit(0);
}
/* Rotation angle is now set to 90 degree. Application can now do
 streaming to see rotated image*/
```

**GIT Method**

```
struct v4l2_control control;

 control.id = V4L2_CID_ROTATION;
 control.value = degree;

ret = ioctl(fd, VIDIOC_S_CTRL, &control);
 if (ret < 0) {
  perror("VIDIOC_S_CTRL\n");
        close(fd);
        exit(0);
    }
```

#### 2.1.6.2. Getting rotation value

PSP1.0.2 release is supporting the rotation through custom ioctls. While git release will support the rotation through the control id. Patch is submitted on to community to add the control id for rotation.

**PSP1.0.x Method**

```
int degree;
ret = ioctl(fd, VIDIOC_G_OMAP2_ROTATION, &degree);
 if (ret < 0) {
  perror("VIDIOC_G_OMAP2_ROTATION\n");
  close(fd);
  exit(0);
 }
/* Degree will contain the rotation value */
```

**GIT Method**

```
struct v4l2_control control;
control.id = V4L2_CID_ROTATION;

ret = ioctl (fd, VIDIOC_G_CTRL, &omap_vout_control);
    if (ret < 0) {
  perror ("VIDIOC_G_CTRL");
  return 0;
 }
printf("%d %d \n", omap_vout_control.id, omap_vout_control.value);
```

#### 2.1.6.3. Querying the control ids supported

PSP1.0.2 release is not supporting any of the control ids. So this ioctl is not applicable for PSP1.0.2 release. Git version currently suppports rotation through control ids. In future mirroring and background cololr setting will be supported throught control ids.

**PSP1.0.x Method**

Not Applicable

**GIT Method**

```
struct v4l2_queryctrl omap_vout_qctrl;
omap_vout_qctrl.id = V4L2_CID_ROTATION;
ret = ioctl (fd, VIDIOC_QUERYCTRL, &omap_vout_qctrl);
 if (ret < 0) {
  perror ("VIDIOC_QUERYCTRL");
  return 0;
 }
printf("%d %s %d %d %d %d %d %d\n",
           omap_vout_qctrl.id,
           omap_vout_qctrl.name,
           omap_vout_qctrl.minimum,
```

```
omap_vout_qctrl.maximum,
omap_vout_qctrl.step,
omap_vout_qctrl.default_value,
omap_vout_qctrl.flags,
omap_vout_qctrl.type);
```

#### 2.1.6.4. Enabling/Disabling the alpha blending

PSP1.0.x was supporting the alpha blending through sysfs entries. PSP02.0x.0x supports the alpha blending through standard V4L2 ioctl. In PSP02.0x.0x release alpha blending will be enabled on the overlay manager on which the video pipeline is currently connected. Suppose vid1 is going to LCD manager and vid2 is going to Digital manager. Then the ioctl on vid1 will enable the alpha blending on LCD manager and ioctl on vid2 will enable alpha blending on Digital manager.

**PSP1.0.x Method**

```
echo on > /sys/class/display_control/omap_disp_control/
lcd_alphablend
 echo off > /sys/class/display_control/omap_disp_control/
lcd_alphablend
 echo on > /sys/class/display_control/omap_disp_control/
tv_alphablend
 echo off > /sys/class/display_control/omap_disp_control/
tv_alphablend
```

**GIT Method**

To Enable the alpha blending on the overlay manager to which the video pipeline is going

```
struct v4l2_framebuffer framebuffer;

ret = ioctl (fd, VIDIOC_G_FBUF, &framebuffer);
if (ret < 0) {
 perror ("VIDIOC_S_FBUF");
 return 0;
}

framebuffer.flags |= V4L2_FBUF_FLAG_LOCAL_ALPHA;
ret = ioctl (fd, VIDIOC_S_FBUF, &framebuffer);
if (ret < 0) {
 perror ("VIDIOC_S_FBUF");
 return 0;
}
```

To Disable the alpha blending on the overlay manager to which the video pipeline is going

```
struct v4l2_framebuffer framebuffer;

ret = ioctl (fd, VIDIOC_G_FBUF, &framebuffer);
```

```
if (ret < 0) {
 perror ("VIDIOC_S_FBUF");
 return 0;
}

framebuffer.flags &= ~V4L2_FBUF_FLAG_LOCAL_ALPHA;
ret = ioctl (fd, VIDIOC_S_FBUF, &framebuffer);
if (ret < 0) {
 perror ("VIDIOC_S_FBUF");
 return 0;
}
```

#### 2.1.6.5. Setting/Getting the global alpha value for video2 pipeline

PSP1.0.x was supporting the global alpha value through sysfs entries. PSP02.0x.0x supports the global alpha value through standard V4L2 ioctl.

**PSP1.0.x Method**

```
echo 255 > /sys/class/display_control/omap_disp_control/
vid2_global_alpha
 echo 0 > /sys/class/display_control/omap_disp_control/
vid2_global_alpha
```

**GIT Method**

Below programlisting shows how to set global alpha value

```
struct v4l2_format fmt;
u8 global_alpha = 128;
fmt.type = V4L2_BUF_TYPE_VIDEO_OVERLAY;

ret = ioctl(fd, VIDIOC_G_FMT, &fmt);
if (ret < 0) {
 perror("VIDIOC_G_FMT\n");
 close(fd);
 exit(0);
}
fmt.fmt.win.global_alpha = global_alpha;

ret = ioctl(fd, VIDIOC_S_FMT, &fmt);
if (ret < 0) {
 perror("VIDIOC_G_FMT\n");
 close(fd);
 exit(0);
}
```

Below programlisting shows how to get global alpha value

```
struct v4l2_format fmt;
fmt.type = V4L2_BUF_TYPE_VIDEO_OVERLAY;

ret = ioctl(fd, VIDIOC_G_FMT, &fmt);
```

```
if (ret < 0) {
 perror("VIDIOC_G_FMT\n");
 close(fd);
 exit(0);
}
printf("Global alpha value read is %d\n",
 fmt.fmt.win.global_alpha);
```

#### 2.1.6.6. Background color.

PSP1.0.x was supporting the background color setting through custom V4L2 ioctl on any of the output device. PSP02.0x.0x supports the background color setting through v4l2 control ioctl. Background color will be set automatically on the overlay manager to which the pipeline is connected.

**PSP1.0.x Method**

Setting the BG Color

```
struct omap24xxvout_bgcolor bgcolor;

bgcolor.color = 0xFF0000; /*Red color */
bgcolor.output_dev = OMAP2_OUTPUT_LCD;

ret = ioctl(fd, VIDIOC_S_OMAP2_BGCOLOR, &bgcolor);
if (ret < 0) {
 perror("VIDIOC_S_OMAP2_BGCOLOR\n");
 close(fd);
 exit(0);
}
```

Getting the BG Color

```
struct omap24xxvout_bgcolor bgcolor;

bgcolor.output_dev = OMAP2_OUTPUT_LCD;

ret = ioctl(fd, VIDIOC_G_OMAP2_BGCOLOR, &bgcolor);
if (ret < 0) {
 perror("VIDIOC_G_OMAP2_BGCOLOR\n");
 close(fd);
 exit(0);
}
printf("Back ground color set is %x\n",bgcolor.color);
```

**GIT Method**

Setting the background color

```
struct v4l2_control control;
unsigned int color = 0xFF0000 /* Red color */

control.id = V4L2_CID_BG_COLOR;
```

```
control.value = color;
ret = ioctl(fd, VIDIOC_S_CTRL, &control);
if (ret < 0) {
 perror("VIDIOC_S_CTRL\n");
 close(fd);
 exit(0);
}
```

Getting the background color

```
struct v4l2_control control;
unsigned int color = 0xFF0000 /* Red color */

control.id = V4L2_CID_BG_COLOR;
ret = ioctl(fd, VIDIOC_G_CTRL, &control);
if (ret < 0) {
 perror("VIDIOC_G_CTRL\n");
 close(fd);
 exit(0);
}
printf("Background color set is %x\n", control.value);
```

#### 2.1.6.7. Enabling/Disabling of the source and destination color keying

PSP1.0.x was supporting the enabling and disabling of the color keying through custom V4L2 ioctl. Color key can be enabled/disabled on any of the output device. While PSP02.0x.0x supports the enabling/disabling of the color keying on the output device to which the video pipeline is connected.

**PSP1.0.x Method (Enabling)**

```
int output_dev = OMAP2_OUTPUT_LCD
ret = ioctl (fd, VIDIOC_OMAP2_COLORKEY_ENABLE, &output_dev);
if (ret < 0) {
 perror ("VIDIOC_OMAP2_COLORKEY_ENABLE");
 return 0;
}
```

**PSP1.0.x Method (Disabling)**

```
int output_dev = OMAP2_OUTPUT_LCD;
ret = ioctl (fd, VIDIOC_OMAP2_COLORKEY_DISABLE, &output_dev);
if (ret < 0) {
 perror ("VIDIOC_OMAP2_COLORKEY_DISABLE");
 return 0;
}
```

**GIT Method**

To Enable the source color keying on the overlay manager to which the video pipeline is connected

```
struct v4l2_framebuffer framebuffer;
```

```
ret = ioctl (fd, VIDIOC_G_FBUF, &framebuffer);
if (ret < 0) {
 perror ("VIDIOC_G_FBUF");
 exit(1);
}
/* Set SRC_COLOR_KEYING if device supports that */
if(framebuffer.capability & V4L2_FBUF_CAP_SRC_CHROMAKEY) {

 framebuffer.flags |= V4L2_FBUF_FLAG_SRC_CHROMAKEY;
 ret = ioctl (fd, VIDIOC_S_FBUF, &framebuffer);
 if (ret < 0) {
  perror ("VIDIOC_S_FBUF");
  exit(1);
 }
}
```

To disable the source color keying on the overlay manager to which the video pipeline is connected

```
struct v4l2_framebuffer framebuffer;
ret = ioctl (fd, VIDIOC_G_FBUF, &framebuffer);
if (ret < 0) {
perror ("VIDIOC_G_FBUF");
exit(1);

}
if(framebuffer.capability & V4L2_FBUF_CAP_SRC_CHROMAKEY) {

 framebuffer.flags >= ~V4L2_FBUF_FLAG_SRC_CHROMAKEY;

 ret = ioctl (fd, VIDIOC_S_FBUF, &framebuffer);
 if (ret < 0) {
  perror ("VIDIOC_S_FBUF");
  exit(1);
 }
}
```

To Enable the destination color keying on the overlay manager to which the video pipeline is connected

```
struct v4l2_framebuffer framebuffer;

ret = ioctl (fd, VIDIOC_G_FBUF, &framebuffer);
if (ret < 0) {
 perror ("VIDIOC_G_FBUF");
 exit(1);
}
/* Set SRC_COLOR_KEYING if device supports that */
if(framebuffer.capability & V4L2_FBUF_CAP_CHROMAKEY) {

 framebuffer.flags |= V4L2_FBUF_FLAG_CHROMAKEY;
 ret = ioctl (fd, VIDIOC_S_FBUF, &framebuffer);
```

```
 if (ret < 0) {
  perror ("VIDIOC_S_FBUF");
  exit(1);
 }
}
```

To disable the source color keying on the overlay manager to which the video pipeline is connected

```
struct v4l2_framebuffer framebuffer;
ret = ioctl (fd, VIDIOC_G_FBUF, &framebuffer);
if (ret < 0) {
perror ("VIDIOC_G_FBUF");
exit(1);

}
if(framebuffer.capability & V4L2_FBUF_CAP_CHROMAKEY) {

 framebuffer.flags &= ~V4L2_FBUF_FLAG_CHROMAKEY;

 ret = ioctl (fd, VIDIOC_S_FBUF, &framebuffer);
 if (ret < 0) {
  perror ("VIDIOC_S_FBUF");
  exit(1);
 }
}
```

#### 2.1.6.8. Setting/Getting the chroma key value

PSP1.0.x was supporting the setting/getting of the chroma key value using the custom ioctl. Chroma key can be set/get on any of the overlay manager. While PSP02.0x.0x supports the setting/getting of the chroma key value using the standard V4L2 ioctl on the overlay manager to which the output device is connected.

**PSP1.0.x Method (Setting)**

```
struct omap24xxvout_colorkey colorkey;
colorkey.outputdev = OMAP2_OUTPUT_LCD;
colorkey.type = OMAP2_GFX_DESTINATION;
colorkey.val = 0xF800; /* Red color RGB565*/

ret = ioctl(fd, VIDIOC_S_OMAP2_COLORKEY, &colorkey);
if (ret < 0) {
 perror("VIDIOC_S_OMAP2_COLORKEY\n");
 close(fd);
 exit(0);
}
```

**PSP1.0.x Method (Getting)**

```
struct omap24xxvout_colorkey colorkey;
```

```
colorkey.outputdev = OMAP2_OUTPUT_LCD;

ret = ioctl(fd, VIDIOC_G_OMAP2_COLORKEY, &colorkey);
if (ret < 0) {
 perror("VIDIOC_S_OMAP2_COLORKEY\n");
 close(fd);
 exit(0);
}
printf("Color key type %d color key value %d\n",
 colorkey.type, colorkey.val);
```

### GIT Method

Below program listing shows how to set the chromakey value using the git method. Here chromakey type is already set during the enablingof the chroma key. Please note that chroma key value should be set before enabling the chroma keying. Overlay manager should not be changed between the setting up of chroma key and enabling the chroma keying.

```
struct v4l2_format fmt;
u8 chromakey = 0xF800; /* Red color RGB565 format */
fmt.type = V4L2_BUF_TYPE_VIDEO_OVERLAY;

ret = ioctl(fd, VIDIOC_G_FMT, &fmt);
if (ret < 0) {
 perror("VIDIOC_G_FMT\n");
 close(fd);
 exit(0);
}
fmt.fmt.win.chromakey = chromakey;

ret = ioctl(fd, VIDIOC_S_FMT, &fmt);
if (ret < 0) {
 perror("VIDIOC_G_FMT\n");
 close(fd);
 exit(0);
}
```

Below programlisting shows how to get global alpha value

```
struct v4l2_format fmt;
fmt.type = V4L2_BUF_TYPE_VIDEO_OVERLAY;

ret = ioctl(fd, VIDIOC_G_FMT, &fmt);
if (ret < 0) {
 perror("VIDIOC_G_FMT\n");
 close(fd);
 exit(0);
}
printf("Global alpha value read is %d\n", fmt.fmt.win.chromakey);
```

# V4L2 Capture Driver

**Abstract**

## 3.1. V4L2 Capture Driver

This document provides the detailed usage of the user interface API changes in the V4L2 Capture driver of GIT release of PSP compared to PSP1.0.2 release. This user interface changes are primarily because of the open source comments and new ISP-Camera library submitted to community.

### 3.1.1. Features Supported

1. One S-video and one Composite SD input in BT.656 format.
2. Supports 8-bit BT.656 capture in UYVY and YUYV interleaved formats.
3. Supports one software channel of capture and a corresponding device node (/dev/video0) is created.
4. Supports single I/O instance and multiple control instances.
5. Supports buffer access mechanism through memory mapping and user pointers.
6. Supports dynamic switching among input interfaces with some necessary restrictions wherever applicable.

7. Supports NTSC and PAL standard on Composite and S-Video interfaces.
8. Supports standard V4L2 IOCTLs to get/set various control parameters like brightness, contrast, saturation, hue and auto gain control.
9. TVP5146 (TVP514x) decoder driver module can be used statically or dynamically (insmod and rmmod supported).

### 3.1.2. Features not supported compared to PSP1.0.2 release

1. The ISP library doesn't configure itself with default params with the detected standard.
2. VIDIOC_G_INPUT doesn't keep on switching the input as in PSP1.0.x.

### 3.1.3. List of IOCTLs same as PSP1.0.2 release

Following APIS/IOCTLs are not changed compare to PSP1.0.2 release. It can be used directly without any changes.

Following is the list of IOCTLs supported in GIT release which are same as PSP1.0.2 release.

- VIDIOC_QUERYBUF
- VIDIOC_REQBUFS
- VIDIOC_QBUF
- VIDIOC_DQBUF
- VIDIOC_STREAMON
- VIDIOC_STREAMOFF
- VIDIOC_G_FMT
- VIDIOC_S_FMT
- VIDIOC_ENUM_FMT
- VIDIOC_TRY_FMT
- VIDIOC_QUERYCAP

### 3.1.4. List of IOCTLs not supported

All the IOCTL's are being supported as compared to PSP1.0.x.

### 3.1.5. List of new IOCTLs in Git release

No new IOCTL's are added as compared to PSP1.0.x release.

## 3.1.6. Usage of New/Changed ioctls in GIT release.

Following section describes the usage of the new/changed ioctls in the Git version of the V4L2 Capture driver compared to PSP1.0.x release.

### 3.1.6.1. VIDIOC_G_INPUT ioctl

Ioctl `VIDIOC_G_INPUT` takes pointer to integer using which the detected inputs will be returned to applicaion space. It will detect all the inputs by calling underneath Video decoder or Sensor interface.

PSP1.0.x driver will return the detected inputs one by one. If there are no further inputs detected, it will start returning the first input detected. So the first call of `VIDIOC_G_INPUT` returns the first input detected, second call returns second input detected and so on until there are no active inputs.

While in GIT release, the ioctl `VIDIOC_G_INPUT` returns only the first active input detected, starting with Composite input.

**PSP1.0.x Method**

```
int index;
struct v4l2_input input;
while (1) {
 if (ioctl(*capture_fd, VIDIOC_G_INPUT, &index) < 0) {
  perror("VIDIOC_G_INPUT");
  goto ERROR;
 }
 input.index = index;
 if (ioctl(*capture_fd, VIDIOC_ENUMINPUT, &input) < 0) {
  perror("VIDIOC_ENUMINPUT");
  goto ERROR;
 }

 printf("Input Detected is: %s\n", input.name);
}
PSP1.0.x:
 - Will keep on changing the input.

GIT2.01.00.0x:
 - Will only detect first active input starting from COMPOSITE.
```