Rockchip DVR&DMS Product Instructions

ID: RK-SM-YF-398

Release Version: V1.1.3

Release Date: 2021-10-11

Security Level: □Top-Secret □Secret □Internal ■Public

DISCLAIMER

Trademark Statement

Rockchip Electronics Co., Ltd.

No.18 Building, A District, No.89, software Boulevard Fuzhou, Fujian,PRC

Website: www.rock-chips.com

Customer service Tel: +86-4007-700-590

Customer service Fax: +86-591-83951833

Customer service e-Mail: fae@rock-chips.com

Preface

Overview

This document is going to introduce DVR&DMS product solution.

Product Version

Chipset	Kernel Version
RV1126, RV1109	Linux 4.19

Intended Audience

This document (this guide) is mainly intended for:

Technical support engineers

Software development engineers

Revision History

Version	Author	Date	Change Description
V1.0.0	Vicent Chi, Zhihua Wang, Zhichao Yu	2021-01-31	Initial version
V1.0.1	Vicent Chi	2021-03-01	Add MPI+DVP solution introduction
V1.0.2	Ruby Zhang	2021-03-15	Update product version
V1.1.0	Zhichao Yu	2021-04-30	Modify the AD chip access solution introduction; Add VP introduction
V1.1.1	Vicent Chi	2021-06-03	Add FAQ
V1.1.2	Zhichao Yu	2021-07-24	Add audio related introduction
V1.1.3	Ruby Zhang	2021-10-11	Fix some expressions

Contents

1. Rockchip DVR/DMS Product Solution Introduction

RV1126 chip has two MIPI interfaces and one DVP interface. In addition, it provides powerful encoding performance and supports up to 8 channels of 1080@15fps simultaneous encoding. With a built-in 2T computing power NPU, it is very suitable for the development of DVR/DMS products.

1.1 Advantages of Developing DVR/DMS Product on RV1126 Platform

- Support up to 8 channels of 1080P analog HD video input;
- Powerful AI processing capabilities, support for operating DMS+ADAS algorithms at the same time;
- Powerful encoding capabilities, up to 8 channels of 1080P@15fps simultaneous encoding;
- Support 8 channels video OSD overlay;
- Support 8 channels video split-screen display demo;

1.2 Analog HD RX Chip Support List

At present, RV1126 platform has been adapted to many analog HD RX chips, and the drivers for these chips have been integrated in the SDK. You can select from the following table:

Model	Manufacturer	Interface	Number of Channels	Maximum Supported Resolution
NVP6188	Nextchip	MIPI	4	4K
N4	Nextchip	MIPI	4	1080P
NVP6158C	Nextchip	DVP	4	2K
TP2815	Techpoint	MIPI	4	1080P
TP2855	Techpoint	MIPI	4	1080P
TP9930	Techpoint	DVP	4	2K
TP9950	Techpoint	MIPI/DVP	1	1080P

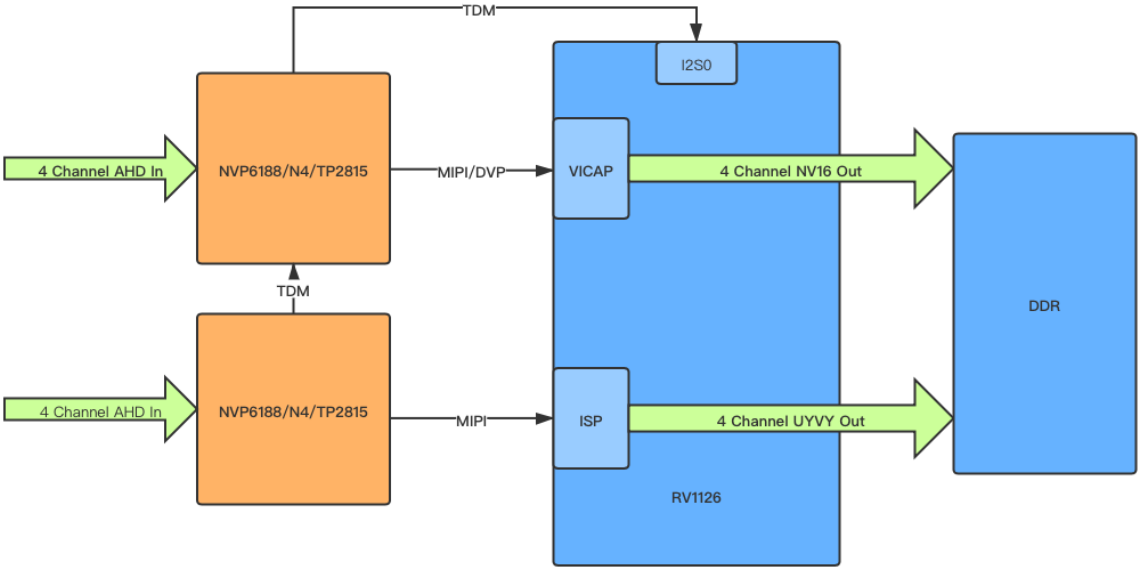
1.3 RV1126 DVR/DMS Products Application Block diagram

RV1126 supports different AD chip access methods, MIPI + MIPI and MIPI + DVP.

Due to the limitation of the RV1126 chip, it is necessary to limit the image output formats of different channels:

- **VICAP channel: all should adopt NV16 format;**
- **ISP access: all should adopt UYVY format;**

The block diagram of the reference solution of camera input is as follows:



2. Analog HD RX Chip Driver Development Introduction

2.1 Kernel Config

Open the relevant config of the RX chip accordingly:

```
CONFIG_VIDEO_NVP6188=y
```

2.2 Kernel dts Configuration

Take NVP6188 RX chip as an example:

```
    nvp6188_0: nvp6188_0@30 {
        compatible = "nvp6188";
        reg = <0x30>;
        clocks = <&cru CLK_MIPICSI_OUT>;
        clock-names = "xvclk";
        power-domains = <&power RV1126_PD_VI>;
        pinctrl-names = "rockchip,camera_default";
        pinctrl-0 = <&mipicsi_clk0>;
        reset-gpios = <&gpio4 RK_PA0 GPIO_ACTIVE_HIGH>;
        power-gpios = <&gpio1 RK_PD4 GPIO_ACTIVE_HIGH>;
        vi-gpios = <&gpio3 RK_PC0 GPIO_ACTIVE_HIGH>;
        rockchip,camera-module-index = <0>;
        rockchip,camera-module-facing = "front";
        rockchip,camera-module-name = "nvp6188";
        rockchip,camera-module-lens-name = "nvp6188";
        port {
            ucam_out0: endpoint {
```

```
                remote-endpoint = <&mipi_in_ucam0>;
                data-lanes = <1 2 3 4>;
            };
        };
    };

    nvp6188_1: nvp6188_1@32 {
        compatible = "nvp6188";
        reg = <0x32>;
        clocks = <&cru CLK_MIPICSI_OUT>;
        clock-names = "xvclk";
        power-domains = <&power RV1126_PD_VI>;
        pinctrl-names = "rockchip,camera_default";
        pinctrl-0 = <&mipicsi_clk1>;
        reset-gpios = <&gpio4 RK_PA1 GPIO_ACTIVE_HIGH>;
        vi-gpios = <&gpio3 RK_PC1 GPIO_ACTIVE_HIGH>;
        rockchip,camera-module-index = <1>;
        rockchip,camera-module-facing = "back";
        rockchip,camera-module-name = "nvp6188";
        rockchip,camera-module-lens-name = "nvp6188";
        port {
            ucam_out1: endpoint {
                remote-endpoint = <&csi_dphy1_input>;
                data-lanes = <1 2 3 4>;
            };
        };
    };

```

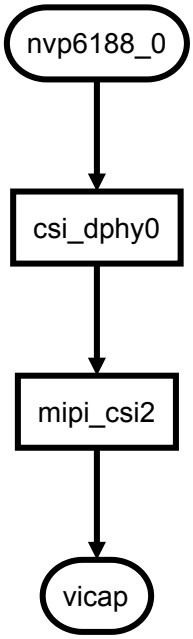
3. Data Streams Channels Introduction

3.1 Two-channel Solution

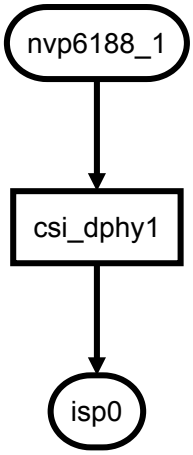
Dual MIPI Solution

Take the dual NVP6188 as an example:

- VICAP channel 0



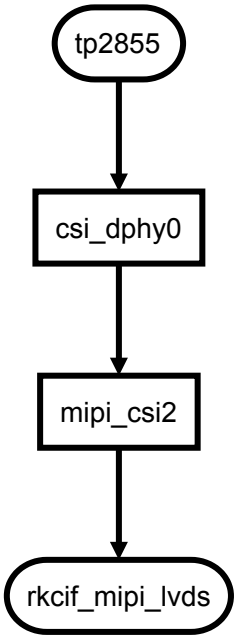
- ISP channel 1



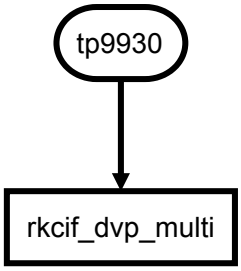
MIPI+DVP solution

Take TP9930+TP2855 for example：

- VICAP Channel 0



- VICAP Channel 1



3.2 Video Format Restrictions of Channels

3.2.1 VICAP Channel

- Use NV16 format for capturing.

3.2.2 ISP Channel

- Use UYVY format for capturing

3.3 Enumeration of Video Nodes of Channels

3.3.1 VICAP Channel

- Capture device node name of stream_cif_mipi_id0/1/2/3 by media-ctl -p -d /dev/mediaX

```

[root@RV1126_RV1109:/]# media-ctl -p -d /dev/media0
Media controller API version 4.19.111

```

```
Media device information
------------------------
driver          rkcif
model           rkcif_mipi_lvds
serial
bus info
hw revision     0x0
driver version  4.19.111

Device topology
- entity 1: stream_cif_mipi_id0 (1 pad, 4 links)
            type Node subtype V4L flags 0
            device node name /dev/video0
        pad0: Sink
                <- "rockchip-mipi-csi2":1 [ENABLED]
                <- "rockchip-mipi-csi2":2 []
                <- "rockchip-mipi-csi2":3 []
                <- "rockchip-mipi-csi2":4 []

- entity 5: stream_cif_mipi_id1 (1 pad, 4 links)
            type Node subtype V4L flags 0
            device node name /dev/video1
        pad0: Sink
                <- "rockchip-mipi-csi2":1 []
                <- "rockchip-mipi-csi2":2 [ENABLED]
                <- "rockchip-mipi-csi2":3 []
                <- "rockchip-mipi-csi2":4 []

- entity 9: stream_cif_mipi_id2 (1 pad, 4 links)
            type Node subtype V4L flags 0
            device node name /dev/video2
        pad0: Sink
                <- "rockchip-mipi-csi2":1 []
                <- "rockchip-mipi-csi2":2 []
                <- "rockchip-mipi-csi2":3 [ENABLED]
                <- "rockchip-mipi-csi2":4 []

- entity 13: stream_cif_mipi_id3 (1 pad, 4 links)
             type Node subtype V4L flags 0
             device node name /dev/video3
        pad0: Sink
                <- "rockchip-mipi-csi2":1 []
                <- "rockchip-mipi-csi2":2 []
                <- "rockchip-mipi-csi2":3 []
                <- "rockchip-mipi-csi2":4 [ENABLED]

```

3.3.2 ISP Channel

- Get device node name of rkisp_mainpath、rkisp_rawwr0/1/2 by media-ctl -p -d /dev/mediaX

```

media-ctl -p -d /dev/media1
Media controller API version 4.19.111

```

```
Media device information
------------------------
driver          rkisp
model           rkisp0
serial
bus info
hw revision     0x0
driver version  4.19.111

Device topology
- entity 17: rkisp_mainpath (1 pad, 1 link)
             type Node subtype V4L flags 0
             device node name /dev/video5
        pad0: Sink
                <- "rkisp-isp-subdev":2 [ENABLED]

- entity 29: rkisp_rawwr0 (1 pad, 1 link)
             type Node subtype V4L flags 0
             device node name /dev/video7
        pad0: Sink
                <- "rkisp-csi-subdev":2 [ENABLED]

- entity 35: rkisp_rawwr1 (1 pad, 1 link)
             type Node subtype V4L flags 0
             device node name /dev/video8
        pad0: Sink
                <- "rkisp-csi-subdev":3 [ENABLED]

- entity 41: rkisp_rawwr2 (1 pad, 1 link)
             type Node subtype V4L flags 0
             device node name /dev/video9
        pad0: Sink
                <- "rkisp-csi-subdev":4 [ENABLED]

```

3.4 Video Capture Limitation of Channels

3.4.1 ISP channel

- pipeline switch

```

    The following code should be added to the boot script(/dev/media1 is determined according to the actual isp registration)

    media-ctl -d /dev/media1 -l '"rkisp-isp-subdev":2->"rkisp-bridge-ispp":0[0]'
    media-ctl -d /dev/media1 -l '"rkisp-isp-subdev":2->"rkisp_mainpath":0[1]'

```

- stream on switch

Because there is no separate switch for the stream on switch of the four channels of rkisp_mainpath and rkisp_rawwr0/1/2, therefore, if you want to capture any one of the rkisp_rawwr0/1/2 three channels, you have to ensure that the three channels will not stream out until the rkisp_mainpath channel is in the stream on state.

- rkisp_mainpath format switch

The output is 1080p by default. If you want to switch the format to 720p, you need to execute:

```
    media-ctl -d /dev/media1 --set-v4l2 '"m01_b_nvp6188 1-
0032":0[fmt:UYVY8_2X8/1280x720]'
    media-ctl -d /dev/media1 --set-v4l2 '"rkisp-csi-
subdev":1[fmt:UYVY8_2X8/1280x720]'
    media-ctl -d /dev/media1 --set-v4l2 '"rkisp-isp-
subdev":0[fmt:YUYV8_2X8/1280x720]'
    media-ctl -d /dev/media1 --set-v4l2 '"rkisp-isp-subdev":0[crop:
(0,0)/1280x720]'
    media-ctl -d /dev/media1 --set-v4l2 '"rkisp-isp-
subdev":2[fmt:YUYV8_2X8/1280x720]'
    media-ctl -d /dev/media1 --set-v4l2 '"rkisp-isp-subdev":2[crop:
(0,0)/1280x720]'

```

3.5 Query Resolution and Video Signal of Channels

- Get subdev node name of sensor by media-ctl -p -d /dev/mediaX

```

media-ctl -p -d /dev/media1
Media controller API version 4.19.111

Media device information
------------------------
driver          rkisp
model           rkisp0
serial
bus info
hw revision     0x0
driver version  4.19.111

Device topology

....

- entity 92: m01_b_nvp6188 1-0032 (1 pad, 1 link)
             type V4L2 subdev subtype Sensor flags 0
             device node name /dev/v4l-subdev6
        pad0: Source
                [fmt:UYVY8_2X8/1920x1080 field:none]
                -> "rockchip-mipi-dphy-rx":0 [ENABLED]


```

- Get the resolution before opening the channel

```c

#include <stdio.h>
#include <unistd.h>
#include <fcntl.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <sys/time.h>
#include <sys/mman.h>
#include <sys/ioctl.h>
#include <linux/videodev2.h>

#define RKMODULE_MAX_VC_CH      4

struct rkmodule_vc_fmt_info {
    __u32 width[RKMODULE_MAX_VC_CH];
    __u32 height[RKMODULE_MAX_VC_CH];
    __u32 fps[RKMODULE_MAX_VC_CH];
} __attribute__ ((packed));

struct rkmodule_vc_hotplug_info {
    __u8 detect_status;
} __attribute__ ((packed));

#define RKMODULE_GET_VC_FMT_INFO \
    _IOR('V', BASE_VIDIOC_PRIVATE + 12, struct rkmodule_vc_fmt_info)

#define RKMODULE_GET_VC_HOTPLUG_INFO \
    _IOR('V', BASE_VIDIOC_PRIVATE + 13, struct rkmodule_vc_hotplug_info)

int main(int argc, char *argv[]) {
    int ch = 0;
    struct rkmodule_vc_hotplug_info status;
    struct rkmodule_vc_fmt_info fmt;
    int fd = open("/dev/v4l-subdev2", O_RDWR, 0);
    ioctl(fd, RKMODULE_GET_VC_FMT_INFO, &fmt);
    ioctl(fd, RKMODULE_GET_VC_HOTPLUG_INFO, &status);
    for(ch = 0; ch < 4; ch++) {
        printf("# ch: %d\n", ch);
        printf("\t width: %d\n", fmt.width[ch]);
        printf("\t height: %d\n", fmt.height[ch]);
        printf("\t fps: %d\n", fmt.fps[ch]);
        printf("\t plug in: %d\n", (status.detect_status & (1 << ch)) ? 1 :
0);
    }
    close(fd);
    return 0;
}

```

3.6 Query Hot Plug Interface In Real Time

- Provide sysfs nodes to user layer for read queries.

```

/sys/devices/platform/ff510000.i2c/i2c-1/1-0032/hotplug_status
/sys/devices/platform/ff510000.i2c/i2c-1/1-0030/hotplug_status

```

4. Multi-channel Audio Capture

The audio of DVR product is generally collected to RV1126 through TDM, and the following commands can be used to collect audio data of a certain channel:

```
arecord  -Dhw:0,0 -c 8 -f S16_LE -r 8000 /tmp/record.wav -vv
```

5. rkmedia_vmix_vo_dvr_test Application Note

The rkmedia_vmix_vo_dvr_test is mainly used to realizes 8-channel video capture and encoding, and 8-channel video synthesis display. The source code is located in SDK/external/rkmedia/examples.

5.1 8-channel Video Capture and H264 Encoding

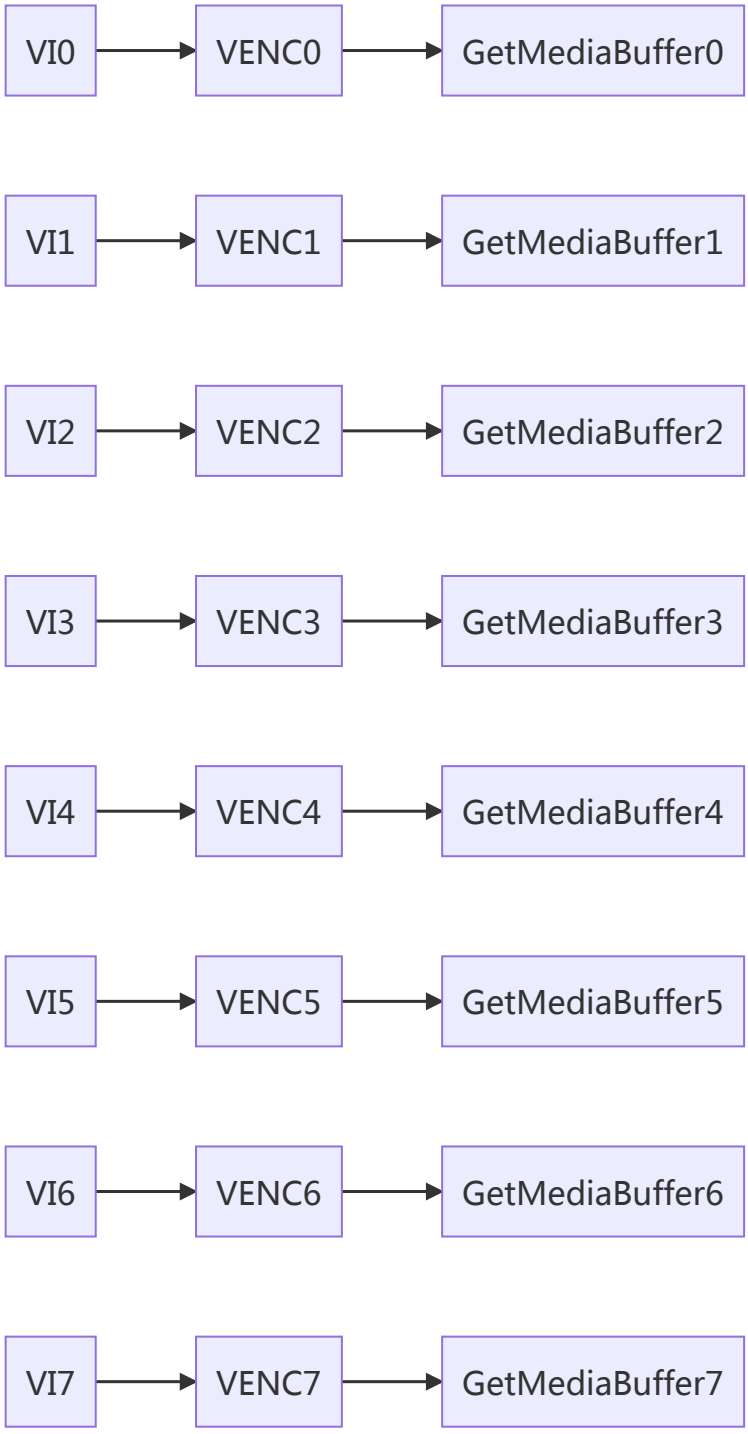
The 8-channel video capture node, resolution, and format are configured through an array, which is convenient for users to modify and debug:

```
stDvr dvr8[8] = {
    {0, "/dev/video30", 1920, 1080, IMAGE_TYPE_NV12, 0},
    {1, "/dev/video31", 1920, 1080, IMAGE_TYPE_NV12, 0},
    {2, "/dev/video32", 1920, 1080, IMAGE_TYPE_NV12, 0},
    {3, "/dev/video33", 1920, 1080, IMAGE_TYPE_NV12, 0},
    {4, "/dev/video37", 1920, 1080, IMAGE_TYPE_NV12, 0},
    {5, "/dev/video38", 1920, 1080, IMAGE_TYPE_NV12, 0},
    {6, "/dev/video39", 1920, 1080, IMAGE_TYPE_NV12, 0},
    {7, "/dev/video40", 1920, 1080, IMAGE_TYPE_NV12, 0},
};
```

According to the recommendation of the dual MIPI solution, it should be modified to:

```
stDvr dvr8[8] = {
    {0, "/dev/video0", 1920, 1080, IMAGE_TYPE_NV16, 0},
    {1, "/dev/video1", 1920, 1080, IMAGE_TYPE_NV16, 0},
    {2, "/dev/video2", 1920, 1080, IMAGE_TYPE_NV16, 0},
    {3, "/dev/video3", 1920, 1080, IMAGE_TYPE_NV16, 0},
    {4, "/dev/video5", 1920, 1080, IMAGE_TYPE_UYVY422, 0},
    {5, "/dev/video7", 1920, 1080, IMAGE_TYPE_UYVY422, 0},
    {6, "/dev/video8", 1920, 1080, IMAGE_TYPE_UYVY422, 0},
    {7, "/dev/video9", 1920, 1080, IMAGE_TYPE_UYVY422, 0},
};
```

The 8-channel video VI implements 8-channel H264 encoding through bind VENC, and the 8-channel VENC-encoded data can be obtained through the GetMediaBuffer thread. Users can realize video transmission requirements base on it.

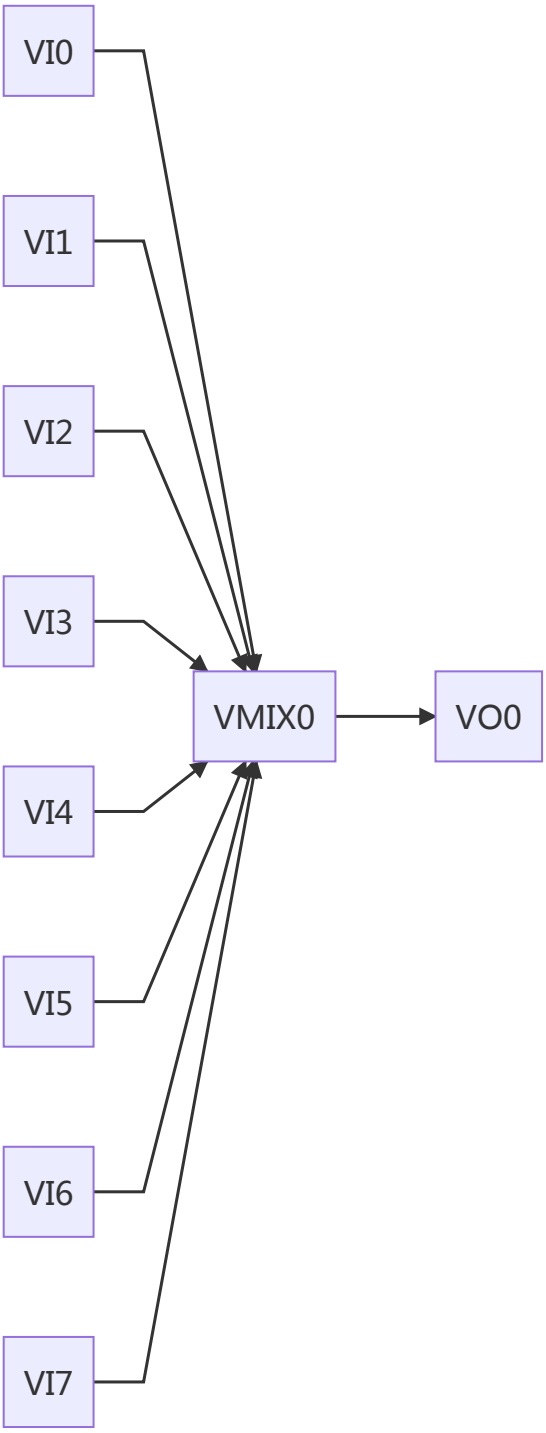


5.2 Support for 8-channel Video Synthesis Display

The 8-channel video specifies the rectangular area of the screen to be displayed through an array, which is convenient for users to modify and debug:

```
RECT_S area_2x4[8] = {
    {0, 0, WIDTH / 2, HEIGHT / 4},
    {WIDTH / 2, 0, WIDTH / 2, HEIGHT / 4},
    {0, HEIGHT / 4, WIDTH / 2, HEIGHT / 4},
    {WIDTH / 2, HEIGHT / 4, WIDTH / 2, HEIGHT / 4},
    {0, HEIGHT / 2, WIDTH / 2, HEIGHT / 4},
    {WIDTH / 2, HEIGHT / 2, WIDTH / 2, HEIGHT / 4},
    {0, HEIGHT * 3 / 4, WIDTH / 2, HEIGHT / 4},
    {WIDTH / 2, HEIGHT * 3 / 4, WIDTH / 2, HEIGHT / 4},
};
```

The 8-channel video synthesis display is realized through the VMIX+VO module:



5.3 Support for 8-channel Video Switching to Front 4-channel and Rear 4-channel Display

Through dvr_bind and dvr_unbind, the 8-channel video can be switched to the front 4-channel and the rear 4-channel display. Users only needs to define the rectangular display area of the front 4-channel and the rear 4-channel.

5.4 Support for Area Frame

Area frame is realized by drawing lines on the whole screen, increasing the area boundary, and specifying the drawing area through an array. The minimum line width is 2, and an even number is required:

```
RECT_S line_2x4[4] = {
    {0, HEIGHT / 4, WIDTH, 2},
    {0, HEIGHT / 2, WIDTH, 2},
    {0, HEIGHT * 3 / 4, WIDTH, 2},
    {WIDTH / 2, 0, 2, HEIGHT},
};
```

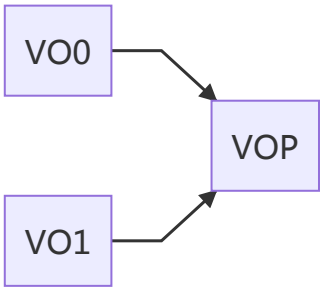
Line drawing area is set through RK_MPI_VMIX_SetLineInfo.

5.5 Support for RGN Cover

It is supported to set sensitive areas for each channel, which is realized through RK_MPI_VMIX_RGN_SetCover.

5.6 Support for Screen OSD

Screen OSD is realized through VO1, users can draw OSD in the buffer (the format is ABGR), after sending to VO1, the OSD can be superimposed on the VO0 video to display the effect through Alpha. In the osd_thread of application, an OSD switching display of two color blocks is drawn every 500ms.



5.7 Support for Channel Display and Hide

Display and hide the channel through RK_MPI_VMIX_ShowChn, RK_MPI_VMIX_HideChn.

5.8 Support for Obtaining Regional Brightness of Channels

Regional brightness of channels is obtained through RK_MPI_VMIX_GetChnRegionLuma, you can get up to 64 regional brightness at a time. The coordinate of each channel is the starting coordinate of the region relative to the channel, not the starting coordinate of the relative screen. You can realize the OSD inverse effect of the screen through the regional brightness.

6. VP Module Introduction

In DVR/DMS products, RGA is used frequently. In order to relieve the pressure of RGA, we use the scaling function of the ISPP module in the RV1126 chip. Therefore, we provide a VP module in rkmedia, through which the scaling function of ISPP can be used.

For details of VP, please refer to the document:
docs/RV1126_RV1109/Multimedia/Rockchip_Developer_Guide_Linux_RKMedia_CN.pdf.

It should be noted that the scaling function of ISPP limits the width of the Buffer and requires 16 Byte alignment.

7. FAQ

7.1 Pictures are Staggered When Hot Plugging

7.1.1 Solution

Add the following configuration in dts to solve the hot plug problem:

```
&rkcif_mipi_lvds {
    ....
    rockchip,cif-monitor = <3 2 25 1000 5>;
    ....
};
```