V316 CSI 模块使用说明书

文档版本号: SDK-V1.0

发布日期: 2019-03-30

上海齐感电子信息科技有限公司senslab

版权所有©珠海全志科技股份有限公司 2019。保留一切权利。
非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

商标声明

、全志和其他全志商标均为珠海全志科技股份有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受全志公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，全志公司对本文档内容不做任何明示或默示的声明或保证。
由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

文档履历

版本号	日期	制/修订人	内容描述
V1.0	2019-03-30	Allwinner	V316 初始化版本

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

目 录

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

表目录

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

图目录

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

1 概述

1.1 编写目的

介绍 VIN（video input）驱动配置，API 接口和上层使用方法

1.2 适用范围

适用 sunxi-vin 模块，基于 linux-4.9 内核

1.3 相关人员

camera 驱动维护人员和应用开发人员

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

2 模块介绍

2.1 模块功能介绍

1. Video input 主要由接口部分（CSI/MIPI）和图像处理单元（ISP/VIPP）组成;

2. CSI/MIPI 部分主要实现视频数据的捕捉;

3. ISP 实现 sensor raw data 数据的处理，包括 lens 补偿、去坏点、gain、gamma、de-mosaic、de-noise、color matrix 等以及一些 3A 的统计;

4. VIPP 能对将图进行缩小、和打水印处理。VIPP 支持 bayer raw data 经过 ISP 处理后再缩小，也支持对一般的 YUV 格式的 sensor 图像直接缩小。

2.2 相关术语介绍

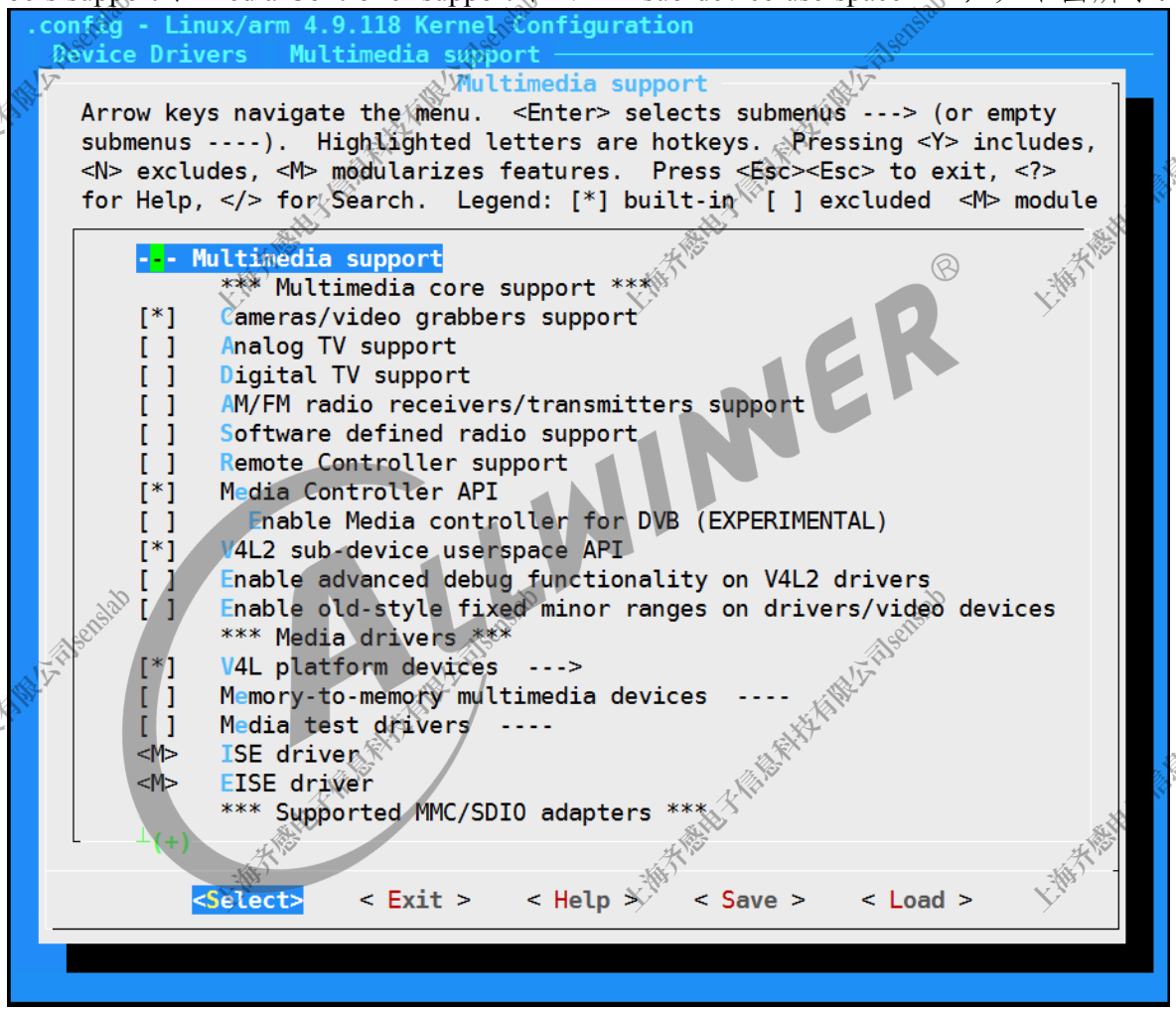
术语	解释说明
ISP	Image Signal Processor 图像信号处理
VIPP	Video Input Post Processor 图像输入后处理
MIPI	Mobile Industry Processor Interface 移动工业处理接口
CCI	Camera Control Interface 摄像头控制接口
MCLK	Master clock（From AP to camera）摄像头主时钟
PCLK	Pixel clock（From camera to AP，Sampling clock for data-bus）像素时钟
YUV	Color Presentation（Y for luminance，U&V for Chrominance）图像数据格式

全志科技版权所有,侵权必究
Copyright © 2019 by Allwinner. All rights reserved

2.3 模块配置介绍

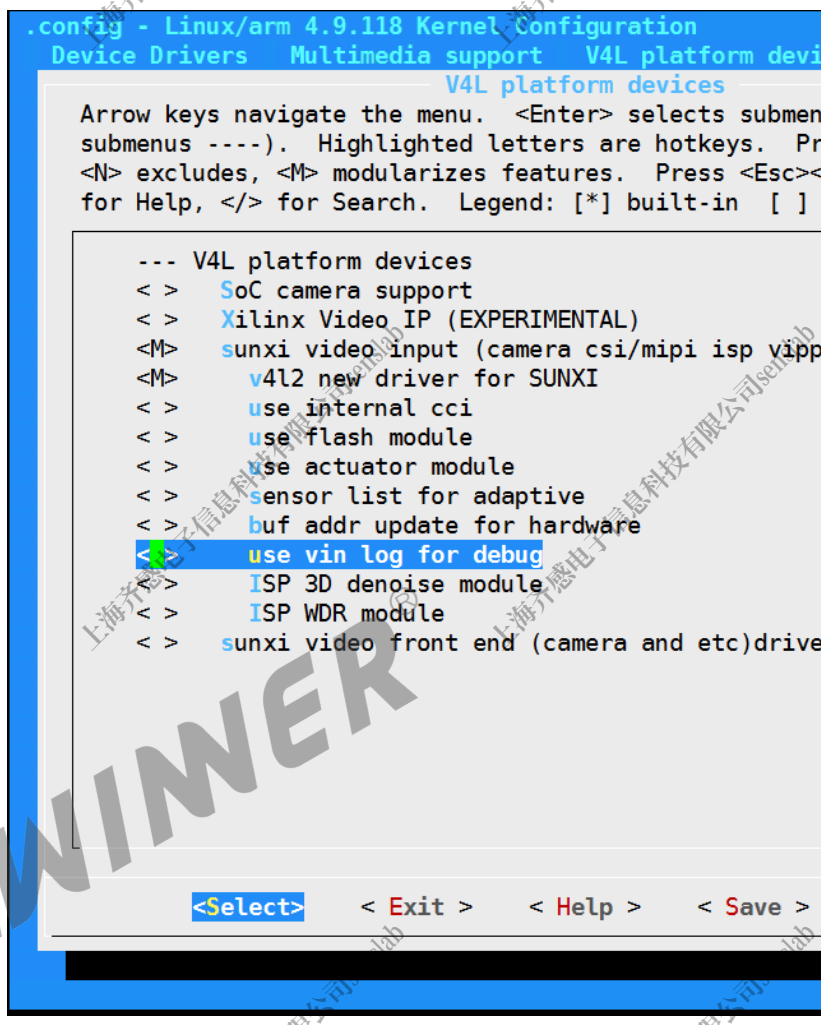
2.3.1 menuconfig 配置说明

1. 首先，进入 Device Drivers，选择 Multimedia support，然后依次打开 Cameras/video grabbers support 、Media Controller support 和 V4L2 sub-device userspace API，如下图所示：



2. 其次，进入 V4L platform devices，选择 sunxi video input (camera csi/mipi isp vipp)driver 和

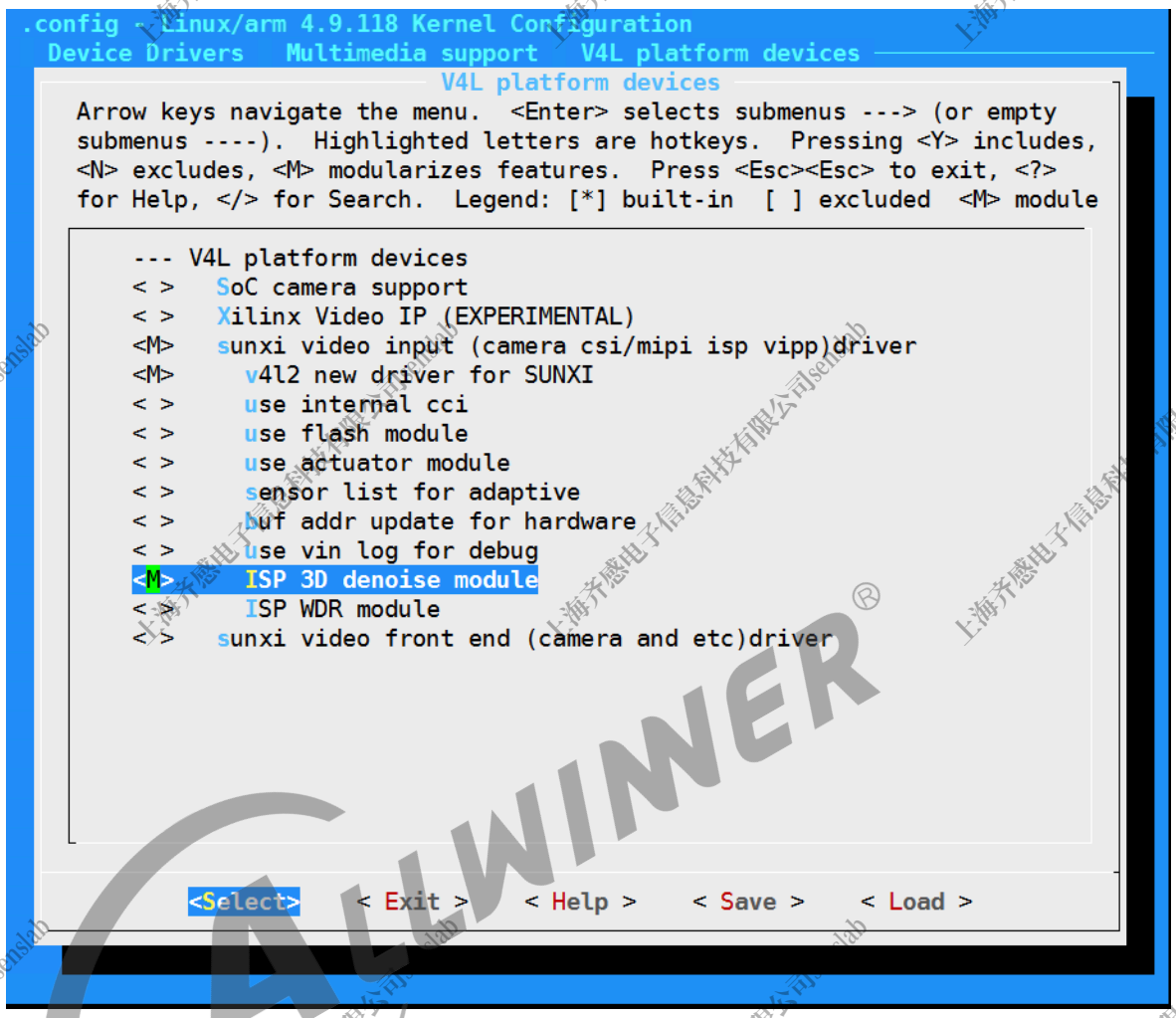
全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved



v4l2 new driver for SUNXI，如下图所示：

3. 最后，sunxi video input (camera csi/mipi isp vipp)driver 目录下的其他选项需要根据实际产品需求进行开关，如：需要用到 ISP 3D 降噪，需要打开 ISP 3D denoise module，如下图

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved



所示:

2.3.2 sys_config.fex 配置说明

Vin 驱动对 sys_config.fex 中 vin 模块的配置做了调整，将 vin 的各个子模块拆开成不通的节点来配置。如下：

```
;vin (video input modules) configuration
;vind(x)_used: 0:disable 1:enable, main node for vin
;csi(x)_used: 0:disable 1:enable
;csi(x)_pck: parallel csi function pin
;csi(x)_mck: parallel csi function pin
;csi(x)_hsync: parallel csi function pin
;csi(x)_vsync: parallel csi function pin
;csi(x)_d0: parallel csi function pin
;csi(x)_d1: parallel csi function pin
;csi(x)_d2: parallel csi function pin
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
;csi(x)_d3: parallel csi function pin
;csi(x)_d4: parallel csi function pin
;csi(x)_d5: parallel csi function pin
;csi(x)_d6: parallel csi function pin
;csi(x)_d7: parallel csi function pin
;flash(x)_used: 0:disable 1:enable
;flash(x)_type: 0:FLASH_RELATING, 1:FLASH_EN_INDEPEND, 2:FLASH_POWER
;flash(x)_en: flash enable gpio, type = 0 of 1
;flash(x)_mode: flash mode gpio, type = 0 of 1
;flash(x)_flvdd: flash module io power handle string, pmu power supply, type = 2
;flash(x)_flvdd_vol: flash module io power voltage, pmu power supply, type = 2
;actuator(x)_used: 0:disable 1:enable
;actuator(x)_name: vcm name
;actuator(x)_slave: vcm iic slave address
;actuator(x)_af_pwdn: vcm power down gpio
;actuator(x)_afvdd: vcm power handle string, pmu power supply
;actuator(x)_afvdd_vol: vcm power voltage, pmu power supply
;sensor(x)_used: 0:disable 1:enable
;sensor(x)_isp_used 0:not use isp 1:use isp
;sensor(x)_fmt: 0:yuv 1:bayer raw rgb
;sensor(x)_stby_mode: 0:not shut down power at standby 1:shut down power at standby
;sensor(x)_vflip: flip in vertical direction 0:disable 1:enable
;sensor(x)_hflip: flip in horizontal direction 0:disable 1:enable
;sensor(x)_iovdd: camera module io power handle string, pmu power supply
;sensor(x)_iovdd_vol: camera module io power voltage, pmu power supply
;sensor(x)_avdd:    camera module analog power handle string, pmu power supply
;sensor(x)_avdd_vol:    camera module analog power voltage, pmu power supply
;sensor(x)_dvdd:    camera module core power handle string, pmu power supply
;sensor(x)_dvdd_vol:    camera module core power voltage, pmu power supply
;sensor(x)_power_en:    camera module power enable gpio
;sensor(x)_reset:   camera module reset gpio
;sensor(x)_pwdn:    camera module pwdn gpio
;fill voltage in uV, e.g. iovdd = 2.8V, sensorx_iovdd_vol = 2800000
;fill handle string as below:
;csi-avdd
;csi-iovdd
;axp22_eldo2
;fill handle string "" when not using any pmu power supply
;vinc(x)_used: 0:disable 1:enable
;vinc(x)_sensor_list: use sensor list
;----------------------------------------------------------------------------------
[vind0]
vind0_used      = 1
vind0_clk       = 300000000
vind0_isp       = 300000000
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
[vind0/csi_cci0]
csi_cci0_used         = 0
csi_cci0_sck          = port:PI05<2><default><default><default>
csi_cci0_sda          = port:PI06<2><default><default><default>

[vind0/csi_cci1]
csi_cci1_used         = 0
csi_cci1_sck          = port:PE16<2><default><default><default>
csi_cci1_sda          = port:PE17<2><default><default><default>

[vind0/csi1]
csi1_used             = 1
csi1_pck              = port:PE00<2><default><default><default>
csi1_hsync            = port:PE02<2><default><default><default>
csi1_vsync            = port:PE03<2><default><default><default>
csi1_d0               = port:PE04<2><default><default><default>
csi1_d1               = port:PE05<2><default><default><default>
csi1_d2               = port:PE06<2><default><default><default>
csi1_d3               = port:PE07<2><default><default><default>
csi1_d4               = port:PE08<2><default><default><default>
csi1_d5               = port:PE09<2><default><default><default>
csi1_d6               = port:PE10<2><default><default><default>
csi1_d7               = port:PE11<2><default><default><default>
csi1_d8               = port:PE12<2><default><default><default>
csi1_d9               = port:PE13<2><default><default><default>
csi1_d10              = port:PE14<2><default><default><default>
csi1_d11              = port:PE15<2><default><default><default>
csi1_d12              = port:PE18<2><default><default><default>
csi1_d13              = port:PE19<2><default><default><default>
csi1_d14              = port:PE20<2><default><default><default>
csi1_d15              = port:PE21<2><default><default><default>

[vind0/flash0]
flash0_used    = 1
flash0_type    = 2
flash0_en      =
flash0_mode    =
flash0_flvdd       = ""
flash0_flvdd_vol   =

[vind0/actuator0]
actuator0_used     = 0
actuator0_name     = "ad5820_act"
actuator0_slave    = 0x18
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
actuator0_af_pwdn      =
actuator0_afvdd        = ”afvcc-csi”
actuator0_afvdd_vol    = 2800000

[vind0/sensor0]
sensor0_used        = 1
sensor0_mname       = ”imx386_mipi”
sensor0_twi_cci_id  = 0
sensor0_twi_addr    = 0x20
sensor0_mclk_id     = 0
sensor0_pos         = ”rear”
sensor0_isp_used    = 1
sensor0_fmt         = 1
sensor0_stby_mode   = 1
sensor0_vflip       = 0
sensor0_hflip       = 0
sensor0_iovdd       = ”iovdd-csi”
sensor0_iovdd_vol   = 1800000
sensor0_avdd        = ”avdd-csi”
sensor0_avdd_vol    = 2800000
sensor0_dvdd        = ””
sensor0_dvdd_vol    = 1500000
sensor0_power_en    =
sensor0_reset       = port:PI3<0><0><1><0>
sensor0_pwdn        = port:PI4<0><0><1><0>
sensor0_sm_hs       = port:PI0<1><0><1><0>
sensor0_sm_vs       = port:PI1<1><0><1><0>

[vind0/sensor1]
sensor1_used        = 0
sensor1_mname       = ”ar0238”
sensor1_twi_cci_id  = 1
sensor1_twi_addr    = 0x20
sensor1_mclk_id     = 1
sensor1_pos         = ”front”
sensor1_isp_used    = 0
sensor1_fmt         = 0
sensor1_stby_mode   = 1
sensor1_vflip       = 0
sensor1_hflip       = 0
sensor1_iovdd       = ”iovdd-csi”
sensor1_iovdd_vol   = 2800000
sensor1_avdd        = ””
sensor1_avdd_vol    = 2800000
sensor1_dvdd        = ””
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
sensor1_dvdd_vol     = 1800000
sensor1_power_en     =
sensor1_reset        =
sensor1_pwdn         =
sensor1_sm_hs        = port:PI0<1><0><1><0>
sensor1_sm_vs        = port:PI1<1><0><1><0>

[vind0/vinc0]
vinc0_used     = 1
vinc0_csi_sel      = 0
vinc0_mipi_sel     = 0
vinc0_isp_sel      = 0
vinc0_isp_tx_ch    = 0
vinc0_rear_sensor_sel   = 0
vinc0_front_sensor_sel  = 0
vinc0_sensor_list   = 0

[vind0/vinc1]
vinc1_used     = 1
vinc1_csi_sel      = 0
vinc1_mipi_sel     = 0
vinc1_isp_sel      = 0
vinc1_isp_tx_ch    = 0
vinc1_rear_sensor_sel   = 0
vinc1_front_sensor_sel  = 0
vinc1_sensor_list   = 0

[vind0/vinc2]
vinc2_used     = 0
vinc2_csi_sel      = 1
vinc2_mipi_sel     = 0xff
vinc2_isp_sel      = 1
vinc2_isp_tx_ch    = 0
vinc2_rear_sensor_sel   = 1
vinc2_front_sensor_sel  = 1
vinc2_sensor_list   = 0

[vind0/vinc3]
vinc3_used     = 0
vinc3_csi_sel      = 1
vinc3_mipi_sel     = 0xff
vinc3_isp_sel      = 1
vinc3_isp_tx_ch    = 0
vinc3_rear_sensor_sel   = 1
vinc3_front_sensor_sel  = 1
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
vinc3_sensor_list  = 0
```

其中：

vind0_usd 是 vin 驱动的总开关，对应的是 media 设备，使用 vin 时必须设为 1；
vind0_clk 是 vin 模块的时钟，实际使用时可以根据 sensor 的帧率和分辨率来设置；vind0_isp 是 isp 模块的时钟，实际使用时可以根据 vind0_clk，sensor 的帧率和分辨率来设置；

csi_cci{x}_used 是 vin 模块中 cci 子模块的开关，当使用系统 IIC 时需要配置为 0，使用 CCI 时配置为 1；
csi_cci{x}_sck 是 cci 模块的 sck 引脚，会与 IIC 复用，所以使用同样引脚的 CCI 与 IIC 不能同时使用；
csi_cci{x}_sda 是 cci 模块的 sda 引脚，会与 IIC 复用，所以使用同样引脚的 CCI 与 IIC 不能同时使用；

csi{x}_used 是 parserx 的使能开关；
csi{x}_xxx 是 dvp（或者其他并行如 BT656 接口）csi 的功能脚，mipi（或者其他串行如 hispi/sublvds）csi 的引脚一般是独占，不需要在 sysconfig 中配置；

flash{x}_used: 0:disable 1:enable
flash{x}_type: 0:FLASH_RELATING, 1:FLASH_EN_INDEPEND, 2:FLASH_POWER
flash{x}_en: flash enable gpio, type = 0 of 1
flash{x}_mode: flash mode gpio, type = 0 of 1
flash{x}_flvdd: flash module io power handle string, pmu power supply, type = 2
flash{x}_flvdd_vol: flash module io power voltage, pmu power supply, type = 2

actuator{x}_used: 0:disable 1:enable
actuator{x}_name: vcm name
actuator{x}_slave: vcm iic slave address
actuator{x}_af_pwdn: vcm power down gpio
actuator{x}_afvdd: vcm power handle string, pmu power supply
actuator{x}_afvdd_vol: vcm power voltage, pmu power supply

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

sensor{x}_used: 0:disable 1:enable
sensor{x}_twi_cci_id：sensor 所使用的 twi 或者 cci 的 id。
sensor{x}_twi_addr：sensor 的 twi 地址
sensor{x}_mclk_id：sensor 所使用的 mclk 的 id。
sensor{x}_pos：sensor 的位置，前置还是后置，主要用在平板上。
sensor{x}_isp_used 0:not use isp 1:use isp
sensor{x}_fmt: 0:yuv 1:bayer raw rgb
sensor{x}_stby_mode: 0:not shut down power at standby 1:shut down power at standby
sensor{x}_vflip: flip in vertical direction 0:disable 1:enable
sensor{x}_hflip: flip in horizontal direction 0:disable 1:enable
sensor{x}_iovdd: camera module io power handle string, pmu power supply
sensor{x}_iovdd_vol: camera module io power voltage, pmu power supply
sensor{x}_avdd: camera module analog power handle string, pmu power supply
sensor{x}_avdd_vol: camera module analog power voltage, pmu power supply
sensor{x}_dvdd: camera module core power handle string, pmu power supply
sensor{x}_dvdd_vol: camera module core power voltage, pmu power supply
sensor{x}_power_en: camera module power enable gpio
sensor{x}_reset: camera module reset gpio
sensor{x}_pwdn: camera module pwdn gpio
flash/actautor/sensor 节点用于对应的外设的开关和配置。这些节点的配置一般需要参考对应方案的原理图和外设的 data sheet 来完成。

vinc{x}_used：vipp 的使能开关，0 — disable，1 — enable。
vinc{x}_csi_sel：表示该 pipeline 上 parser 的 id，必须配置，且为有效 id。
vinc{x}_mipi_sel：表示该 pipeline 上 mipi（sublvds/hispi）的 id，不使用时配置为 0xff。
vinc{x}_isp_sel：表示该 pipeline 上 isp 的 id，必须配置，当 isp 为空时，这个 isp 只是表示路由不做 isp 的效果处理。
vinc{x}_isp_tx_ch 表示该 pipeline 上 isp 的 ch，必须配置，默认为 0。当 sensor 是 bt656 多通道或者 WDR 出 RAW 时，该 ch 可以配置 0 ～ 3 的值。
vinc{x}_rear_sensor_sel 表示该 pipeline 上使用的后置 sensor 的 id。
vinc{x}_front_sensor_sel 表示该 pipeline 上使用的前置 sensor 的 id。
vinc{x}_sensor_list 表示是否使用 sensor_list 来时适配不同的模组，1 表示使用，0 表示不使用。

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

2.4 源码结构介绍

驱动路径位于 drivers/media/platform/sunxi-vin 目录

```
sunxi-vin:. |
  vin.c          ;设备驱动media |
  vin.h |
  Kconfig |
  Makefile |
  top_reg.c          ; 硬件底层实现top |
  top_reg.h          ;底层实现头文件top |
  top_reg_i.h        ;寄存器资源头文件top | ├──

modules |
  ├──actuator |
  |       actuator.c          ; vcm 的一般行为driver |
  |       actuator.h          ; vcm 的头文件driver |
  |       ad5820_act.c |
  |       dw9714_act.c |
  |       Makefile |
  |       ov8825_act.c |
  | |
  ├──flash |
  |       flash.c          ; flash 的源文件driver |
  |       flash.h |
  | |
  ├──sensor |
  |       camera.h         ; 公用结构体头文件camera |
  |       camera_cfg.h     ;camera 扩展命令头文件ioctl |
  |       gc2155.c |
  |       gc5004.c |
  |       Makefile |
  |       ov2710_mipi.c |
  |       ov4689.c |
  |       ov8858.c |
  |       ov5640.c |
  |       sensor_helper.h   ; 帮助函数头文件sensor |
  |       sensor_helper.c   ; 公共函数源文件sensor |
  |       sensor_spi.c      ;spi 驱动模板sensor |
  | | ├──

platform |
    platform_cfg.h          平台配置头文件; |
```

全志科技版权所有, 侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
    sun8iw12p1_vin_cfg.h  具体的平台配置头文件;|
    sun8iw16p1_vin_cfg.h|
    sun50iw3p1_vin_cfg.h| ├──

utility|
   bsp_common.c        通过像素类型获取信息的实现函数;|
   bsp_common.h        通过像素类型获取信息的头文件;|
   cfg_op.c           读取;文件的实现函数ini|
   cfg_op.h           读取;文件函数对应的头文件ini|
   config.c           读取设备树实现函数;|
   config.h           读取设备树头文件;|
   vin_io.h           ;模块寄存器操作头文件vin|
   vin_os.c           操作;状态和申请内存源文件GPIO|
   vin_os.h           操作;状态和申请内存头文件GPIO|
   vin_supply.c         ;供电和时钟操作源文件sensor|
   vin_supply.h         ;供电和时钟操作头文件sensor| ├──


vin-cci|
    bsp_cci.c          底层;cci 函数bsp|
    bsp_cci.h          底层;cci 函数头文件bsp|
    cci_helper.c        ;cci 帮助函数，供驱动调用sensor|
    cci_helper.h        ;cci 帮助函数头文件|
    csi_cci_reg.c       ;硬件底层实现cci|
    csi_cci_reg.h       ;硬件底层实现头文件cci|
    csi_cci_reg_i.h      ;cci 寄存器资源头文件|
    sunxi_cci.c        ;cci 平台驱动源文件|
    sunxi_cci.h        ;cci 平台驱动头文件| ├──

vin-csi|
    parser_reg.c       ;硬件底层实现parseri|
    parser_reg.h       ;硬件底层实现头文件parser|
    parser_reg_i.h      ;parser 寄存器资源头文件|
    sunxi_csi.c        ;csi 子模块驱动源文件|
    sunxi_csi.h        ;csi 子模块驱动头文件| ├──

vin-isp|
    isp_default_tbl.h    ;isp 表配置头文件table|
    sunxi_isp.c       ;子模块驱动源文件isp|
    sunxi_isp.h       ;子模块驱动头文件isp ├──
isp520|
  |    isp520_reg.c     ;isp5版底层实现函数.20|
  |    isp520_reg.h     ;isp5版底层实现函数头文件.20|
  |    isp520_reg_i.h    ;isp5版寄存器资源头文件.20 ├──
sun8iw12p1|
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
  |      sun8iw12p1_reg.c    ;isp5版底层实现函数.10 |
  |      sun8iw12p1_reg.h    ;isp5版底层实现函数头文件.10 |
  |      sun8iw12p1_reg_i.h    ;isp5版寄存器资源头文件.10 | ├──

vin-mipi |
    bsp_mipi_csi.c          底层;mipi 函数bsp |
    bsp_mipi_csi.h          底层;mipi 函数头文件bsp |
    bsp_mipi_csi_null.c      底层;mipi 空函数bsp |
    bsp_mipi_csi_v1.c       底层;mipi bsp 版函数v1 |
    combo_common.h       ;口公共资源头文件combo |
    protocol.h            ;mipi 协议层头文件 |
  ├──dphy |
  |      dphy.h        ;mipi 头文件dphy |
  |      dphy_reg.c     ;mipi 底层实现函数dphy |
  |      dphy_reg.h     ;mipi 底层实现函数头文件dphy |
  |      dphy_reg_i.h    ;mipi dphy 寄存器资源头文件 |
  | |
  └──protocol |
  |      protocol.h      ;mipi 协议层头文件 |
  |      protocol_reg.c    ;mipi 协议层底层实现 |
  |      protocol_reg.h    ;mipi 协议层底层实现头文件 |
  |      protocol_reg_i.h   ;mipi 协议层寄存器资源头文件 |
  | |
  └──combo_rx |
      combo_rx_reg.c    ;combo 协议层底层实现 |
      combo_rx_reg.h    ;combo 协议层底层实现头文件 |
      combo_rx_reg_i.h   ;combo 协议层寄存器资源头文件 |
    sunxi_mipi.c         ;mipi 子模块驱动源文件 |
    sunxi_mipi.h         ;mipi 子模块驱动头文件 ├──
vin-scaler |
    sunxi_scaler.c |
    sunxi_scaler.h |
    vipp_reg.c        ;底层实现vipp |
    vipp_reg.h        ;底层实现头文件vipp |
    vipp_reg_i.h      ;寄存器资源头文件vipp | ├──

vin-stat |
    vin_h3a.c         直方图和;3统计驱动abuf |
    vin_h3a.h |
    vin_ispstat.c      统计;管理框架buf |
    vin_ispstat.h | ├──

vin-video |
    vin_core.c        ;资源获取，中断处理video |
    vin_core.h |
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
vin_video.c      ;节点操作集和回调videovideobuf |
vin_video.h |
dma_reg.c        ;底层实现dma |
dma_reg.h        ;底层实现头文件dma |
dma_reg_i.h      ;寄存器资源头文件dma └──
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

3 接口描述

3.1 VIDIOC_QUERYCAP

3.1.1 Parameters

```
Capability of csi （driverstruct v4l2_capability * ） capability
struct v4l2_capability {
    __u8    driver[16]; /* i.e. ”bttv” */
    __u8    card[32];   /* i.e. ”Hauppauge WinTV” */
    __u8    bus_info[32];   /* ”PCI:” + pci_name(pci_dev) */
    __u32   version;        /* should use KERNEL_VERSION() */
    __u32   capabilities;   /* Device capabilities */
    __u32   reserved[4];
};
```

3.1.2 Returns

Success:0; Fail: Failure Number ### Description 获取驱动的名称、版本、支持的 capabilities 等，如 V4L2_CAP_STREAMING，V4L2_BUF_TYPE_VIDEO_CAPTURE_MPLANE 等

3.2 VIDIOC_ENUM_INPUT

3.2.1 Parameters

```
（
inputstruct v4l2_input *）inp
struct v4l2_input {
    __u32       index;    /*  Which input */
    __u8        name[32];     /*  Label */
    __u32       type;     /*  Type of input */
    __u32       audioset;     /*  Associated audios (bitfield) */
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
    __u32       tuner;          /* Associated tuner */
    v4l2_std_id  std;
    __u32       status;
    __u32       capabilities;
    __u32       reserved[3];
};
```

3.2.2 Returns

Success:0; Fail: Failure Number ### Description 获取驱动支持的 input index。目前驱动只支持 input index = 0 或 index = 1。

Index = 0 表示 primary csi device

Index = 1 表示 secondary csi device

应用输入 index 参数，驱动返回 type。对于 VIN 设备来说，type 为 V4L2_INPUT_TYPE_CAMERA。

3.3 VIDIOC_S_INPUT

3.3.1 Parameters

```
（
inputstruct v4l2_input *）inp
The same as VIDIOC_ENUM_INPUT
```

3.3.2 Returns

Success:0; Fail: Failure Number ### Description 通过 inp.index 设置当前要访问的 csi device 为 primary device 还是 secondary device。

Index = 0 （双摄像头配置中，一般对应后置双摄像头。若只有一个摄像头设备，则 index 固定为 0）

Index = 1 （双摄像头配置中，一般对应前置摄像头）

调用该接口后，实际上会对 csi device 进行初始化工作。

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

3.4 VIDIOC_G_INPUT

3.4.1 Parameters

```
（
inputstruct v4l2_input *）inp
The same as VIDIOC_ENUM_INPUT
```

3.4.2 Returns

Success:0; Fail: Failure Number ### Description 获取 inp.index，判断当前设置的 csi device 为 primary device 还是 secondary device。

Index = 0 （双摄像头配置中，一般对应后置双摄像头。若只有一个摄像头设备，则 index 固定为 0）

Index = 1（双摄像头配置中，一般对应前置摄像头）

3.5 VIDIOC_S_PARM

3.5.1 Parameters

```
（
Parameterstruct v4l2_streamparm *）parms
struct v4l2_streamparm {
   enum v4l2_buf_type type;
   union {
      struct v4l2_captureparm capture;
      struct v4l2_outputparm  output;
      __u8    raw_data[200];  /* user-defined */
   } parm;
};

struct v4l2_captureparm {
   __u32         capability;   /*  Supported modes */
   __u32         capturemode;  /*  Current mode */
   struct v4l2_fract  timeperframe;  /*  Time per frame in .1us units */
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
    __u32       extendedmode;  /*  Driver-specific extensions */
    __u32           readbuffers;   /*  # of buffers for read */
    __u32       reserved[4];
};
```

3.5.2 Returns

Success:0; Fail: Failure Number ### Description CSI 作为输入设备，只关注 parms.type 和 parms. capture。
应用使用时，parms.type = V4L2_BUF_TYPE_VIDEO_CAPTURE_MPLANE；
其中通过设定 parms->capture.capturemode（V4L2_MODE_VIDEO 或 V4L2_MODE_IMAGE），实现视频或图片的采集。通过设定 parms->capture.timeperframe，可以设置帧率。

3.6 VIDIOC_G_PARM

3.6.1 Parameters

```
（
Parameterstruct v4l2_streamparm *）parms
The same as VIDIOC_S_PARM
```

3.6.2 Returns

Success:0; Fail: Failure Number ### Description 应用使用时，parms.type = V4L2_BUF_TYPE_VIDEO_CAP
通过 parms->capture.capturemode 返回当前是 V4L2_MODE_VIDEO 或 V4L2_MODE_IMAGE；
通过 parms->capture.timeperframe, 返回当前设置的帧率

3.7 VIDIOC_ENUM_FMT

3.7.1 Parameters

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
V4L2 （formatstruct v4l2_fmtdesc * ） fmtdesc
struct v4l2_fmtdesc {
    __u32           index;          /* Format number      */
    enum v4l2_buf_type  type;           /* buffer type        */
    __u32               flags;
    __u8            description[32];   /* Description string */
    __u32           pixelformat;      /* Format fourcc      */
    __u32           reserved[4];
};
```

3.7.2 Returns

Success:0; Fail: Failure Number ### Description 获取驱动支持的 V4L2 格式

应用输入 type，index 参数，驱动返回 pixelformat 。对于 VIN 设备来说，type 为 V4L2_BUF_TYPE_VIDEO_CAPTURE_MPLANE。

3.8 VIDIOC_TRY_FMT

3.8.1 Parameters

```
Video type, format and （sizestruct v4l2_format * ） fmt
struct v4l2_format {
    enum v4l2_buf_type type;
    union {
        struct v4l2_pix_format      pix;
        struct v4l2_pix_format_mplane   pix_mp;
        struct v4l2_window      win;
        struct v4l2_vbi_format      vbi;
        struct v4l2_sliced_vbi_format   sliced;
        __u8    raw_data[200];
    } fmt;
};

struct v4l2_pix_format {
    __u32               width;
    __u32           height;
    __u32           pixelformat;
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
    enum v4l2_field     field;
    __u32               bytesperline;   /* for padding, zero if unused */
    __u32               sizeimage;
    enum v4l2_colorspace    colorspace;
    __u32           priv;       /* private data, depends on pixelformat */
};
```

3.8.2 Returns

Success:0; Fail: Failure Number ### Description 根据捕捉视频的类型、格式和大小，判断模式、格式等是否被驱动支持。不会改变任何硬件设置。

对于 VIN 设备，type 为 V4L2_BUF_TYPE_VIDEO_CAPTURE_MPLANE。使用 struct v4l2_pix_format_mplane 进行参数传递。

应用程序输入 struct v4l2_pix_format_mplane 结构体里面的 width、height、pixelformat、field 等参数，驱动返回最接近的 width、height；若 pixelformat、field 不支持，则默认选择驱动支持的第一种格式。

3.9 VIDIOC_S_FMT

3.9.1 Parameters

```
Video type, format and （sizestruct v4l2_format * ） fmt
The same as VIDIOC_TRY_FMT
```

3.9.2 Returns

Success:0; Fail: Failure Number ### Description 设置捕捉视频的类型、格式和大小，设置之前会调用 VIDIOC_TRY_FMT。

对于 VIN 设备，type 为 V4L2_BUF_TYPE_VIDEO_CAPTURE_MPLANE。使用 struct v4l2_pix_format_mplane 进行参数传递。

应用程序输入 width、height、pixelformat、field 等，驱动返回最接近的 width、height；若 pixelformat、field 不支持，则默认选择驱动支持的第一种格式。

应用程序应该以驱动返回的 width、height、pixelformat、field 等作为后续使用传递的参数。

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

对于 OSD 设备，type 为 V4L2_BUF_TYPE_VIDEO_OVERLAY。使用 struct v4l2_window 进行参数传递。

应用程序输入水印的个数、窗口位置和大小、bitmap 地址、bitmap 格式以及 global_alpha 等。驱动保存这些参数，并在 VIDIOC_OVERLAY 命令传递使能命令时生效。

3.10 VIDIOC_G_FMT

3.10.1 Parameters

Video type, format and （sizestruct v4l2_format * ）fmt The same as VIDIOC_TRY_FMT

3.10.2 Returns

Success:0; Fail: Failure Number ### Description 获取捕捉视频的 width、height、pixelformat、field、bytesperline、sizeimage 等参数

3.11 VIDIOC_OVERLAY

3.11.1 Parameters

Overlay on/（offunsigned int ）i

3.11.2 Returns

Success:0; Fail: Failure Number ###Description 传递 1 表示使能，0 表示关闭。设置使能时会更新 osd 参数，使之生效。

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

3.12 VIDIOC_REQBUFS

3.12.1 Parameters

```
Buffer type ,count and memory map （typestruct v4l2_requestbuffers * ）req
struct v4l2_requestbuffers {
    __u32           count;
    enum v4l2_buf_type      type;
    enum v4l2_memory        memory;
    __u32           reserved[2];
};
```

3.12.2 Returns

Success:0; Fail: Failure Number ### Description v4l2_requestbuffers 结构中定义了缓存的数量，驱动会据此申请对应数量的视频缓存。多个缓存可以用于建立 FIFO，来提高视频采集的效率。这些 buffer 通过内核申请，申请后需要通过 mmap 方法，映射到 User 空间。

Count：定义需要申请的 video buffer 数量

Type：对于 VIN 设备，为 V4L2_BUF_TYPE_VIDEO_CAPTURE_MPLANE

Memory：目前支持 V4L2_MEMORY_MMAP、V4L2_MEMORY_USERPTR、V4L2_MEMORY_DMABUF 方式

应用程序传递上述三个参数，驱动会根据 VIDIOC_S_FMT 设置的格式计算供需要 buffer 的大小，并返回 count 数量。

3.13 VIDIOC_QUERYBUF

3.13.1 Parameters

```
Buffer type ,index and memory map （typestruct v4l2_buffer *）buf
struct v4l2_buffer {
    __u32           index;
    enum v4l2_buf_type      type;
    __u32           bytesused;
    __u32           flags;
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
    enum v4l2_field     field;
    struct timeval      timestamp;
    struct v4l2_timecode    timecode;
    __u32           sequence;

    /* memory location */
    enum v4l2_memory        memory;
    union {
        __u32       offset;
        unsigned long   userptr;
        struct v4l2_plane *planes;
    } m;
    __u32           length;
    __u32           input;
    __u32           reserved;
};
```

3.13.2 Returns

Success:0; Fail: Failure Number ### Description 通过 struct v4l2_buffer 结构体的 index，访问对应序号的 buffer，获取到对应 buffer 的缓存信息。主要利用 length 信息及 m.offset 信息来完成 mmap 操作。

3.14 VIDIOC_DQBUF

3.14.1 Parameters

```
Buffer type ,index and memory map （typestruct v4l2_buffer *）buf
struct v4l2_buffer is the same as VIDIOC_QUERYBUF
```

3.14.2 Returns

Success:0; Fail: Failure Number ### Description 将 driver 已经填充好数据的 buffer 出列，供应用使用。

应用程序根据 index 来识别 buffer，此时 m.offset 表示 buffer 对应的物理地址。

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

3.15 VIDIOC_QBUF

3.15.1 Parameters

Buffer type ,index and memory map （typestruct v4l2_buffer *）buf

3.15.2 Returns

Success:0; Fail: Failure Number ### Description 将 User 空间已经处理过的 buffer，重新入队，移交给 driver，等待填充数据。
应用程序根据 index 来识别 buffer。

3.16 VIDIOC_STREAMON

3.16.1 Parameters

Buffer （typeenum v4l2_buf_type *）type

3.16.2 Returns

Success:0; Fail: Failure Number ### Description 此处的 buffer type 为 V4L2_BUF_TYPE_VIDEO_CAPTURE
运行此 IOCTL，将 buffer 队列中所有 buffer 入队，并开启 CSIC DMA 硬件中断，每次中断便表示完成一帧 buffer 数据的填入。

3.17 VIDIOC_STREAMOFF

3.17.1 Parameters

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
Buffer （typeenum v4l2_buf_type *）type
```

3.17.2 Returns

Success:0; Fail: Failure Number ### Description 此处的 buffer type 为 V4L2_BUF_TYPE_VIDEO_CAPTURE 运行此 IOCTL，停止捕捉视频，将 frame buffer 队列清空，以及 video buffer 释放。

3.18 VIDIOC_QUERYCTRL

3.18.1 Parameters

```
Control id and （valuestruct v4l2_queryctrl *）qc
struct v4l2_queryctrl {
    __u32          id;
    enum v4l2_ctrl_type  type;
    __u8           name[32];  /* Whatever */
    __s32          minimum;   /* Note signedness */
    __s32          maximum;
    __s32          step;
    __s32          default_value;
    __u32          flags;
    __u32          reserved[2];
};
```

3.18.2 Returns

Success:0; Fail: Failure Number ### Description 应用程序通过 id 参数，驱动返回需要调节参数的 name，minmum，maximum，default_value 以及步进 step。（由 v4l2 conctrols framework 完成）

目前可能支持的 id 请参考 VIDIOC_S_CTRL

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

3.19 VIDIOC_S_CTRL

3.19.1 Parameters

Control id and （valuestruct v4l2_queryctrl *）qc The same as VIDIOC_QUERYCTRL

3.19.2 Returns

Success:0; Fail: Failure Number ### Description 应用程序通过 id，value 等参数，对 camera 驱动对应的参数进行设置。
驱动内部会先调用 vidioc_queryctrl，判断 id 是否支持，value 是否在 minimum 和 maximum 之间。（由 v4l2 conctrols framework 完成）
目前可能支持的 id 和 value 参考附件。

3.20 VIDIOC_G_CTRL

3.20.1 Parameters

Control id and （valuestruct v4l2_queryctrl *）qc The same as VIDIOC_QUERYCTRL

3.20.2 Returns

Success:0; Fail: Failure Number ### Description 应用程序通过 id，驱动返回对应 id 当前设置的 value。

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

3.21 VIDIOC_ENUM_FRAMESIZES

3.21.1 Parameters

```
index,type,（formatstruct ）v4l2_frmsizeenum

enum v4l2_frmsizetypes {
    V4L2_FRMSIZE_TYPE_DISCRETE  = 1,
    V4L2_FRMSIZE_TYPE_CONTINUOUS  = 2,
    V4L2_FRMSIZE_TYPE_STEPWISE  = 3,
};

struct v4l2_frmsize_discrete {
    __u32          width;      /* Frame width [pixel] */
    __u32          height;     /* Frame height [pixel] */
};

struct v4l2_frmsize_stepwise {
    __u32          min_width;  /* Minimum frame width [pixel] */
    __u32          max_width;  /* Maximum frame width [pixel] */
    __u32          step_width; /* Frame width step size [pixel] */
    __u32          min_height; /* Minimum frame height [pixel] */
    __u32          max_height; /* Maximum frame height [pixel] */
    __u32          step_height;    /* Frame height step size [pixel] */
};

struct v4l2_frmsizeenum {
    __u32          index;      /* Frame size number */
    __u32          pixel_format;  /* Pixel format */
    __u32          type;       /* Frame size type the device supports. */

    union {                /* Frame size */
        struct v4l2_frmsize_discrete    discrete;
        struct v4l2_frmsize_stepwise    stepwise;
    };

    __u32   reserved[2];           /* Reserved space for future use */
};
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

3.21.2 Returns

Success:0; Fail: Failure Number ### Description 根据应用传进来的 index，pixel_format，驱动返回 type，并根据 type 填写 discrete 或 stepwise 的值。Discrete 表示分辨率固定的值；stepwise 表示分辨率有最小值和最大值，并根据 step 递增。上层根据返回的 type，做对应不同的操作。

3.22 VIDIOC_ENUM_FRAMEINTERVALS

3.22.1 Parameters

```
, , , （
Indexformatsizetypestruct ）v4l2_frmivalenum

enum v4l2_frmivaltypes {
    V4L2_FRMIVAL_TYPE_DISCRETE  = 1,
    V4L2_FRMIVAL_TYPE_CONTINUOUS    = 2,
    V4L2_FRMIVAL_TYPE_STEPWISE  = 3,
};

struct v4l2_frmival_stepwise {
    struct v4l2_fract   min;        /* Minimum frame interval [s] */
    struct v4l2_fract   max;        /* Maximum frame interval [s] */
    struct v4l2_fract   step;       /* Frame interval step size [s] */
};

struct v4l2_frmivalenum {
    __u32           index;      /* Frame format index */
    __u32           pixel_format;   /* Pixel format */
    __u32           width;      /* Frame width */
    __u32           height;     /* Frame height */
    __u32           type;       /* Frame interval type the device supports. */

    union {                 /* Frame interval */
        struct v4l2_fract       discrete;
        struct v4l2_frmival_stepwise    stepwise;
    };

    __u32   reserved[2];            /* Reserved space for future use */
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
};
```

3.22.2 Returns

Success:0; Fail: Failure Number ### Description 应用程序通过 pixel_format、width、height、驱动返回 type，并根据 type 填写 V4L2_FRMIVAL_TYPE_DISCRETE、V4L2_FRMIVAL_TYPE_CONTINUOU 或 V4L2_FRMIVAL_TYPE_STEPWISE。Discrete 表示支持单一的帧率；stepwise 表示支持步进的帧率。

3.23 VIDIOC_ISP_EXIF_REQ

作用: 得到当前照片的 EXIF 信息，填写到相应的编码域中。目的：对于 raw sensor 尽量填写正规的 EXIF 信息，yuv sensor 该 IOCTRL 也可以使用，不过驱动中填写的也是固定值。相关参数：

```
struct v4l2_fract {
    __u32   numerator;
    __u32   denominator;
};

struct isp_exif_attribute {
    struct v4l2_fract exposure_time;
    struct v4l2_fract shutter_speed;
    __u32 aperture;
    __u32 focal_length;
    __s32 exposure_bias;
    __u32 iso_speed;
    __u32 flash_fire;
    __u32 brightness;
};

struct v4l2_fract exposure_time;曝光时间：分数类型，例如
numerator = ，1denominator = ，则表示秒的曝光时间。2001/200

struct v4l2_fract shutter_speed;快门速度：分数类型，例如
numerator = ，1denominator = ，则表示秒的快门速度。（实际上和曝光时间数值相同）2001/200

__u32 aperture;光圈大小：，例如
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

FNumberaperture = ，则表示，光圈大小为，即222.2FNumber = 22/10;

__u32 focal_length;焦距：例如
focal_length = ，则表示焦距为140014，即mmFocalLength = 1400/100(mm);

__s32 exposure_bias;曝光补偿：范围
-4~4

__u32 iso_speed;感光速度：
50~3200

__u32 flash_fire;闪光灯是否开启：
flash_fire = 1 表示闪光灯开启，flash_fire = 0 表示闪光灯未开启。

__u32 brightness;图像亮度：
0~255.使用示例：

```
int V4L2CameraDevice::getExifInfo(struct isp_exif_attribute *exif_attri)
{
    int ret = -1;
    if (mCameraFd == NULL)
    {
        return 0xFF000000;
    }
    ret = ioctl(mCameraFd, VIDIOC_ISP_EXIF_REQ, exif_attri);
    return ret;
}
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

4 调用流程

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

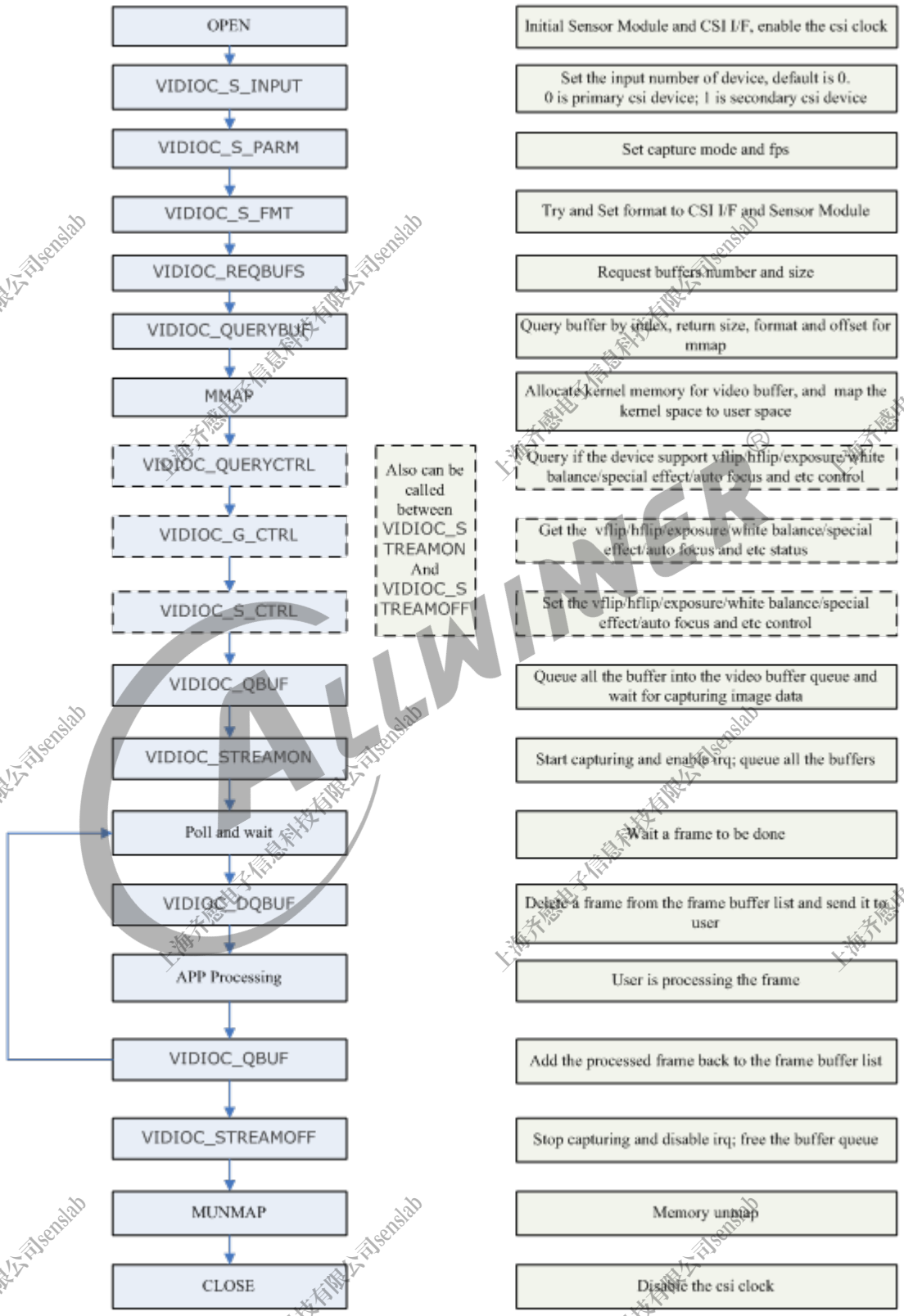


图 4-1: CSI 调用流程

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

5 demo

如下代码为 v4l2 API 的基本调用 demo，主要实现 camera 视频流格式，大小的设置，以及怎样获取 frame buffer 和释放。

```
/*
 * zw
 * for csi & isp test
 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <assert.h>
#include <time.h>

#include <getopt.h>

#include <fcntl.h>
#include <unistd.h>
#include <errno.h>
#include <malloc.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <sys/time.h>
#include <sys/mman.h>
#include <sys/ioctl.h>

#include <asm/types.h>

#include "../sunxi_camera_v2.h"

#define CLEAR(x) (memset(&(x), 0, sizeof (x)))
#define ALIGN_4K(x) (((x) + (4095)) & ~(4095))
#define ALIGN_16B(x) (((x) + (15)) & ~(15))

struct size {
    int width;
    int height;
};
struct buffer {
    void *start[3];
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
    int length[3];
};

static char path_name[20];
static char dev_name[20];
static int fd = -1;
static int      isp0_fd     = -1;
static int      isp1_fd     = -1;

struct buffer *buffers;
static unsigned int n_buffers;

struct size input_size;

unsigned int req_frame_num = 8;
unsigned int read_num = 20;
unsigned int count;
unsigned int nplanes;

int buf_size;

static int read_frame(int mode)
{
    struct v4l2_buffer buf;
    char fdstr[30];
    FILE *file_fd = NULL;

    CLEAR(buf);
    buf.type = V4L2_BUF_TYPE_VIDEO_CAPTURE_MPLANE;
    buf.memory = V4L2_MEMORY_MMAP;
    buf.length = nplanes;
    buf.m.planes =
        (struct v4l2_plane *)calloc(nplanes, sizeof(struct v4l2_plane));

    if (-1 == ioctl(fd, VIDIOC_DQBUF, &buf)) {
        free(buf.m.planes);
        return -1;
    }

    assert(buf.index < n_buffers);

    if (count == read_num / 2) {
        printf("file length = %d %d %d\n", buffers[buf.index].length[0],
                buffers[buf.index].length[1],
                buffers[buf.index].length[2]);
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
        printf("file start = %p %p %p\n", buffers[buf.index].start[0],
                buffers[buf.index].start[1],
                buffers[buf.index].start[2]);

        sprintf(fdstr, "%s/fb%d_y%d.bin", path_name, 1, mode);
        file_fd = fopen(fdstr, "w");
        fwrite(buffers[buf.index].start[0], buffers[buf.index].length[0], 1, file_fd);
        fclose(file_fd);

        sprintf(fdstr, "%s/fb%d_u%d.bin", path_name, 1, mode);
        file_fd = fopen(fdstr, "w");
        fwrite(buffers[buf.index].start[1], buffers[buf.index].length[1], 1, file_fd);
        fclose(file_fd);

        sprintf(fdstr, "%s/fb%d_v%d.bin", path_name, 1, mode);
        file_fd = fopen(fdstr, "w");
        fwrite(buffers[buf.index].start[2], buffers[buf.index].length[2], 1, file_fd);
        fclose(file_fd);
    }

    if (-1 == ioctl(fd, VIDIOC_QBUF, &buf)) {
        free(buf.m.planes);
        return -1;
    }

    free(buf.m.planes);
    return 0;
}

static int req_frame_buffers(void)
{
    unsigned int i;
    struct v4l2_requestbuffers req;

    CLEAR(req);
    req.count = req_frame_num;
    req.type = V4L2_BUF_TYPE_VIDEO_CAPTURE_MPLANE;
    req.memory = V4L2_MEMORY_MMAP;
    if (-1 == ioctl(fd, VIDIOC_REQBUFS, &req)) {
        printf("VIDIOC_REQBUFS error\n");
        return -1;
    }
    buffers = calloc(req.count, sizeof(*buffers));

    for (n_buffers = 0; n_buffers < req.count; ++n_buffers) {
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
    struct v4l2_buffer buf;
    CLEAR(buf);
    buf.type = V4L2_BUF_TYPE_VIDEO_CAPTURE_MPLANE;
    buf.memory = V4L2_MEMORY_MMAP;
    buf.index = n_buffers;
    buf.length = nplanes;
    buf.m.planes =
      (struct v4l2_plane *)calloc(nplanes,
              sizeof(struct v4l2_plane));
    if (NULL == buf.m.planes) {
      printf("buf.m.planes calloc failed!\n");
      return -1;
    }
    if (-1 == ioctl(fd, VIDIOC_QUERYBUF, &buf)) {
      printf("VIDIOC_QUERYBUF error\n");
      free(buf.m.planes);
      return -1;
    }

    for (i = 0; i < nplanes; i++) {
      buffers[n_buffers].length[i] = buf.m.planes[i].length;
      buffers[n_buffers].start[i] =
        mmap(NULL /* start anywhere */ ,
         buf.m.planes[i].length,
         PROT_READ | PROT_WRITE /* required */,
         MAP_SHARED /* recommended */ ,
         fd, buf.m.planes[i].m.mem_offset);

      if (MAP_FAILED == buffers[n_buffers].start[i]) {
        printf("mmap failed\n");
        free(buf.m.planes);
        return -1;
      }
    }
    free(buf.m.planes);
  }

  for (i = 0; i < n_buffers; ++i) {
    struct v4l2_buffer buf;
    CLEAR(buf);
    buf.type = V4L2_BUF_TYPE_VIDEO_CAPTURE_MPLANE;
    buf.memory = V4L2_MEMORY_MMAP;
    buf.index = i;
    buf.length = nplanes;
    buf.m.planes =
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
        (struct v4l2_plane *)calloc(nplanes,
                sizeof(struct v4l2_plane));

    if (-1 == ioctl(fd, VIDIOC_QBUF, &buf)) {
        printf("VIDIOC_QBUF failed\n");
        free(buf.m.planes);
        return -1;
    }
    free(buf.m.planes);
  }
  return 0;
}

static int free_frame_buffers(void)
{
  unsigned int i, j;

  for (i = 0; i < n_buffers; ++i) {
    for (j = 0; j < nplanes; j++)
      if (-1 ==
        munmap(buffers[i].start[j], buffers[i].length[j])) {
        printf("munmap error");
        return -1;
      }
  }
  free(buffers);
  return 0;
}

static int subdev_open(int *sub_fd, char *str)
{
  char subdev[20] = {'\0'};
  char node[50] = {'\0'};
  char data[20] = {'\0'};
  int i, fs = -1;

  for (i = 0; i < 255; i++) {
    sprintf(node, "/sys/class/video4linux/v4l-subdev%d/name", i);
    fs = open(node, O_RDONLY/* required */| O_NONBLOCK, 0);
    if (fs < 0) {
      printf("open %s falied\n", node);
      continue;
    }
    /*data_length = lseek(fd, 0, SEEK_END);*/
    lseek(fs, 0L, SEEK_SET);
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
        read(fs, data, 20);
        close(fs);
        if (!strncmp(str, data, strlen(str))) {
            sprintf(subdev, "/dev/v4l-subdev%d", i);
            printf("find %s is %s\n", str, subdev);
            *sub_fd = open (subdev, O_RDWR | O_NONBLOCK, 0);
            if (*sub_fd < 0) {
                printf("open %s falied\n", str);
                return -1;
            }
            printf("open %s fd = %d\n", str, *sub_fd);
            return 0;
        }
    }
    printf("can not find %s!\n", str);
    return -1;
}

static int camera_init(int sel, int mode)
{
    struct v4l2_input inp;
    struct v4l2_streamparm parms;

    fd = open(dev_name, O_RDWR /* required */ | O_NONBLOCK, 0);

    if (fd < 0) {
        printf("open falied\n");
        return -1;
    }
    printf("open %s fd = %d\n", dev_name, fd);

    if (-1 == subdev_open(&isp0_fd, "sunxi_isp.0"))
        return -1;
    if (-1 == subdev_open(&isp1_fd, "sunxi_isp.1"))
        return -1;

    inp.index = sel;
    if (-1 == ioctl(fd, VIDIOC_S_INPUT, &inp)) {
        printf("VIDIOC_S_INPUT %d error!\n", sel);
        return -1;
    }
    parms.type = V4L2_BUF_TYPE_VIDEO_CAPTURE_MPLANE;
    parms.parm.capture.timeperframe.numerator = 1;
    parms.parm.capture.timeperframe.denominator = 30;/*fps*/
    parms.parm.capture.capturemode = V4L2_MODE_VIDEO;
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
    /*when different video have the same sensor source*/
    parms.parm.capture.reserved[0] = 0; /*1:use sensor current win, 0:find the nearest win*/
    parms.parm.capture.reserved[1] = 0; /*2:command, 1: wdr, 0: normal*/

    if (-1 == ioctl(fd, VIDIOC_S_PARM, &parms)) {
        printf("VIDIOC_S_PARM error\n");
        return -1;
    }

    return 0;
}

static int camera_fmt_set(int subch, int angle)
{
    struct v4l2_format fmt;

    CLEAR(fmt);
    fmt.type = V4L2_BUF_TYPE_VIDEO_CAPTURE_MPLANE;
    fmt.fmt.pix_mp.width = input_size.width;
    fmt.fmt.pix_mp.height = input_size.height;
    fmt.fmt.pix_mp.pixelformat = V4L2_PIX_FMT_YUV420;
    fmt.fmt.pix_mp.field = V4L2_FIELD_NONE;

    if (-1 == ioctl(fd, VIDIOC_S_FMT, &fmt)) {
        printf("VIDIOC_S_FMT error!\n");
        return -1;
    }

    if (-1 == ioctl(fd, VIDIOC_G_FMT, &fmt)) {
        printf("VIDIOC_G_FMT error!\n");
        return -1;
    } else {
        nplanes = fmt.fmt.pix_mp.num_planes;
        printf("resolution got from sensor = %d*%d num_planes = %d\n",
            fmt.fmt.pix_mp.width, fmt.fmt.pix_mp.height,
            fmt.fmt.pix_mp.num_planes);
    }
    return 0;
}

static int main_test(int sel, int mode)
{
    enum v4l2_buf_type type = V4L2_BUF_TYPE_VIDEO_CAPTURE_MPLANE;
    struct v4l2_ext_control ctrls[4];
    struct v4l2_ext_controls ext_ctrls;
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
int i;

int subch = 0;
int angle = 0;

if (mode >= 1) {
    subch = 1;
    if (mode == 2)
        angle = 90;
    else
        angle = 270;
}

if (-1 == camera_init(sel, mode))
    return -1;
if (-1 == camera_fmt_set(subch, angle))
    return -1;
if (-1 == req_frame_buffers())
    return -1;

if (-1 == ioctl(fd, VIDIOC_STREAMON, &type)) {
    printf("VIDIOC_STREAMON failed\n");
    return -1;
} else
    printf("VIDIOC_STREAMON ok\n");

count = read_num;
while (count-- > 0) {
    for (;;) {
        fd_set fds;
        struct timeval tv;
        int r;

        FD_ZERO(&fds);
        FD_SET(fd, &fds);

        tv.tv_sec = 2;  /* Timeout. */
        tv.tv_usec = 0;

        for (i = 0; i < 4; i++) {
            ctrls[i].id = V4L2_CID_R_GAIN + i;
            ctrls[i].value = count % 256;
        }
        memset(&ext_ctrls, 0, sizeof ext_ctrls);
        ext_ctrls.ctrl_class = V4L2_CID_R_GAIN;
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
    ext_ctrls.count = 4;
    ext_ctrls.controls = ctrls;
    ioctl (isp0_fd, VIDIOC_S_EXT_CTRLS, &ext_ctrls);

    for (i = 0; i < 4; i++) {
        ctrls[i].id = V4L2_CID_AE_WIN_X1 + i;
        ctrls[i].value = count*16 % 256;
    }
    memset(&ext_ctrls, 0, sizeof ext_ctrls);
    ext_ctrls.ctrl_class = V4L2_CID_AE_WIN_X1;
    ext_ctrls.count = 4;
    ext_ctrls.controls = ctrls;
    ioctl (isp0_fd, VIDIOC_S_EXT_CTRLS, &ext_ctrls);

    for (i = 0; i < 4; i++) {
        ctrls[i].id = V4L2_CID_AF_WIN_X1 + i;
        ctrls[i].value = count*16 % 256;
    }
    memset(&ext_ctrls, 0, sizeof ext_ctrls);
    ext_ctrls.ctrl_class = V4L2_CID_AF_WIN_X1;
    ext_ctrls.count = 4;
    ext_ctrls.controls = ctrls;
    ioctl (isp0_fd, VIDIOC_S_EXT_CTRLS, &ext_ctrls);
    r = select(fd + 1, &fds, NULL, NULL, &tv);

    if (-1 == r) {
        if (EINTR == errno)
            continue;
        printf("select err\n");
    }
    if (0 == r) {
        fprintf(stderr, "select timeout\n");
        if (-1 == ioctl(fd, VIDIOC_STREAMOFF, &type))
            printf("VIDIOC_STREAMOFF failed\n");
        else
            printf("VIDIOC_STREAMOFF ok\n");
        free_frame_buffers();
        return -1;
    }

    if (!read_frame(mode))
        break;
    else
        return -1;
}
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
    }

    if (-1 == ioctl(fd, VIDIOC_STREAMOFF, &type)) {
        printf("VIDIOC_STREAMOFF failed\n");
        return -1;
    } else
        printf("VIDIOC_STREAMOFF ok\n");

    if (-1 == free_frame_buffers())
        return -1;

    close(isp0_fd);
    close(isp1_fd);
    return 0;
}

int main(int argc, char *argv[])
{
    int i, test_cnt = 1;
    int sel = 0;
    int width = 640;
    int height = 480;
    int mode = 1;

    CLEAR(dev_name);
    CLEAR(path_name);
    if (argc == 1) {
        sprintf(dev_name, "/dev/video0");
        sprintf(path_name, "/mnt/sdcard");
    } else if (argc == 3) {
        sel = atoi(argv[1]);
        sprintf(dev_name, "/dev/video%d", sel);
        sel = atoi(argv[2]);
        sprintf(path_name, "/mnt/sdcard");
    } else if (argc == 5) {
        sel = atoi(argv[1]);
        sprintf(dev_name, "/dev/video%d", sel);
        sel = atoi(argv[2]);
        width = atoi(argv[3]);
        height = atoi(argv[4]);
        sprintf(path_name, "/mnt/sdcard");
    } else if (argc == 6) {
        sel = atoi(argv[1]);
        sprintf(dev_name, "/dev/video%d", sel);
        sel = atoi(argv[2]);
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
        width = atoi(argv[3]);
        height = atoi(argv[4]);
        sprintf(path_name, "%s", argv[5]);
    } else if (argc == 7) {
        sel = atoi(argv[1]);
        sprintf(dev_name, "/dev/video%d", sel);
        sel = atoi(argv[2]);
        width = atoi(argv[3]);
        height = atoi(argv[4]);
        sprintf(path_name, "%s", argv[5]);
        mode = atoi(argv[6]);
    } else if (argc == 8) {
        sel = atoi(argv[1]);
        sprintf(dev_name, "/dev/video%d", sel);
        sel = atoi(argv[2]);
        width = atoi(argv[3]);
        height = atoi(argv[4]);
        sprintf(path_name, "%s", argv[5]);
        mode = atoi(argv[6]);
        test_cnt = atoi(argv[7]);
    } else {
        printf("please select the video device: 0-video0 1-video1 ......\n");
        scanf("%d", &sel);
        sprintf(dev_name, "/dev/video%d", sel);

        printf("please select the camera: 0-dev0 1-dev1 ......\n");
        scanf("%d", &sel);

        printf("please input the resolution: width height......\n");
        scanf("%d %d", &width, &height);

        printf("please input the frame saving path......\n");
        scanf("%15s", path_name);

        printf("please input the test mode: 0~3......\n");
        scanf("%d", &mode);

        printf("please input the test_cnt: >=1......\n");
        scanf("%d", &test_cnt);
    }

    input_size.width = width;
    input_size.height = height;
    buf_size = ALIGN_16B(input_size.width) * input_size.height * 3 / 2;
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

```
    for (i = 0; i < test_cnt; i++) {
        if (0 == main_test(sel, mode))
            printf("mode %d test done at the %d time!!\n",
                mode, i);
        else
            printf("mode %d test failed at the %d time!!\n",
                mode, i);
        close(fd);
    }
    return 0;
}
```

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved

6 Declaration

This document is the original work and copyrighted property of Allwinner Technology (‘“Allwinner”‘). Reproduction in whole or in part must obtain the written approval of Allwinner and give clear acknowledgement to the copyright owner.

The information furnished by Allwinner is believed to be accurate and reliable. Allwinner reserves the right to make changes in circuit design and/or specifications at any time without notice. Allwinner does not assume any responsibility and liability for its use. Nor for any infringements of patents or other rights of the third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of Allwinner. This datasheet neither states nor implies warranty of any kind, including fitness for any particular application.tates nor implies warranty of any kind, including fitness for any particular application.

全志科技版权所有，侵权必究
Copyright © 2019 by Allwinner. All rights reserved