cfg\_type--->

ISPIF\_CLK\_ENABLE,

ISPIF\_CLK\_DISABLE,

ISPIF\_INIT,//初始化

ISPIF\_CFG,//配置

ISPIF\_START\_FRAME\_BOUNDARY,

ISPIF\_RESTART\_FRAME\_BOUNDARY,

ISPIF\_STOP\_FRAME\_BOUNDARY,

ISPIF\_STOP\_IMMEDIATELY,

ISPIF\_RELEASE,

ISPIF\_ENABLE\_REG\_DUMP,

ISPIF\_SET\_VFE\_INFO,

ISPIF\_CFG2,

vfe设置

设置数据：

ispif\_cfg\_data：

1：类型：cfg\_type

2：reg\_dump

3：csid\_version

4：vfe\_info

5：params

vfe\_info:

msm\_ispif\_vfe\_info:

--->int num\_vfe;//vfe的数量

--->struct msm\_isp\_info info[VFE\_MAX];};

--->uint32\_t max\_resolution;

uint32\_t id;

uint32\_t ver;

Params

msm\_ispif\_param\_data

--->uint32\_t num;

--->struct msm\_ispif\_params\_entry entries[MAX\_PARAM\_ENTRIES];

--->enum msm\_ispif\_vfe\_intf vfe\_intf;//主要有VFE0,VFE1

enum msm\_ispif\_intftype intftype;//主要有PIX0,RDI0,PIX1,RDI1,RDI2

每个vfe都有自己的一个pix0和3个rdi

int num\_cids;

enum msm\_ispif\_cid cids[MAX\_CID\_CH\_v2];//

cid值与虚拟通道匹配：

虚拟通道0：可以选择的cid:0,1,2,3

虚拟通道1：可以选择的cid:4,5,6,7

虚拟通道2：可以选择的cid:8,9,10,11

虚拟通道3：可以选择的cid:12,13,14,15//每个虚拟通道最多4个cid

enum msm\_ispif\_csid csid;

8953 3个csid:CSID0,CSID1,CSID2

int crop\_enable;//使能剪切

uint16\_t crop\_start\_pixel;

uint16\_t crop\_end\_pixel;

camera

Isp图像处理

内存。

rdi通道直接绕过图像处理管道

pxi经由图像处理管道。

Vfe0

Vfe1

模块包含图像处理硬件块的管道。VFE有不同的输入接口。PIX (Pixel)输入接口将输入数据提供给图像处理管道。图像处理管道的末尾还包含一个缩放和裁剪模块。三个RDI(原始转储接口)输入接口绕过图像处理管道。VFE还包含一个AXI总线接口，用于将输出数据写入内存。

ISPIF

处理数据流从csid路由到VFE输入的过程

4中通路：PIX0,RDI0,RDI1,RDI2,RDI3

Csid0

Csid1

Csid2//设置虚拟通道

它们处理CSI2接收机的协议和应用层。CSID可以解码来自任何CSIPHY的数据流

Csi\_phy0

Csi\_phy1

Csi\_phy2

处理CSI2接收器的物理层。一个单独的摄像头传感器可以连接到每一个CSIPHY模块

msm\_ispif\_subdev\_ioctl\_unlocked

--->case VIDIOC\_MSM\_ISPIF\_CFG:

--->return msm\_ispif\_cmd(sd, arg);

--->case ISPIF\_CFG:

--->msm\_ispif\_config(ispif, &pcdata->params);

设置：

根据intftype，vfe\_intf选择csid并使能csid

如根据rdi1,vfe0,csid0

--->msm\_ispif\_sel\_csid\_core(ispif, intftype,

params->entries[i].csid, vfe\_intf);

--->cid\_mask = msm\_ispif\_get\_cids\_mask\_from\_cfg(

&params->entries[i]);

--->msm\_ispif\_enable\_intf\_cids(ispif, intftype,

cid\_mask, vfe\_intf, 1);

iface\_ispif\_proc\_streamon

--->设置ispif的参数

--->iface\_ispif\_get\_cfg\_params\_from\_hw\_streams(iface, iface\_sink\_port,

session, num\_hw\_streams, hw\_stream\_ids, cfg\_cmd, start);

--->cfg\_cmd->cfg\_type = ISPIF\_CFG;

--->ioctl(ispif->fd, VIDIOC\_MSM\_ISPIF\_CFG, cfg\_cmd);

--->iface\_ispif\_pack\_config(iface, session,

num\_hw\_streams, hw\_stream\_ids, cfg\_cmd);

--->

ISPIF的设置

kernel

config\_proc\_get\_stream\_bit

--->switch (image\_mode) {

case MSM\_V4L2\_EXT\_CAPTURE\_MODE\_DEFAULT:

case MSM\_V4L2\_EXT\_CAPTURE\_MODE\_PREVIEW:

return V4L2\_DEV\_STRAEMON\_BIT\_P;

case MSM\_V4L2\_EXT\_CAPTURE\_MODE\_VIDEO:

return V4L2\_DEV\_STRAEMON\_BIT\_V;

case MSM\_V4L2\_EXT\_CAPTURE\_MODE\_THUMBNAIL:

return V4L2\_DEV\_STRAEMON\_BIT\_T;

case MSM\_V4L2\_EXT\_CAPTURE\_MODE\_MAIN:

return V4L2\_DEV\_STRAEMON\_BIT\_S;

case MSM\_V4L2\_EXT\_CAPTURE\_MODE\_ISP\_PIX\_OUTPUT1:

return V4L2\_DEV\_STREAMON\_BIT\_ISP\_OUT1;

case MSM\_V4L2\_EXT\_CAPTURE\_MODE\_ISP\_PIX\_OUTPUT2:

return V4L2\_DEV\_STREAMON\_BIT\_ISP\_OUT2;

case MSM\_V4L2\_EXT\_CAPTURE\_MODE\_RAW:

return V4L2\_DEV\_STREAMON\_BIT\_R;

case MSM\_V4L2\_EXT\_CAPTURE\_MODE\_RDI:

return V4L2\_DEV\_STREAMON\_BIT\_RDI;

case MSM\_V4L2\_EXT\_CAPTURE\_MODE\_RDI1:

return V4L2\_DEV\_STREAMON\_BIT\_RDI1;

case MSM\_V4L2\_EXT\_CAPTURE\_MODE\_RDI2:

return V4L2\_DEV\_STREAMON\_BIT\_RDI2;

default:

return 0;

ispif\_client\_set\_params

设置channel\_interface\_mask，进而配置ISPIF\_CFG

ispif\_client\_get\_params

--->ispif\_client = get\_ispif\_client\_info(handle);

--->case ISPIF\_SET\_INTF\_PARAMS:

--->uint32\_t intf\_mask = ispif\_client->channel\_interface\_mask;

--->uint32\_t stream\_mask = ispif\_set->data.channel\_stream\_info;

config\_MSM\_V4L2\_STREAM\_ON

设置

case MSM\_V4L2\_CAM\_OP\_CAPTURE:

config\_update\_channel\_stream\_mask(ctrl, ctrlCmd->stream\_type, 1);

其中ctrlCmd->stream\_type决定pvideo\_ctrl->streamon\_bundle\_mask |=

config\_proc\_get\_stream\_bit(ctrlCmd->stream\_type)---->确认使用的，根据image mode确认所使用的

Rdi raw isp

config\_update\_channel\_stream\_mask

--->根据rdi,isp模式设置：

ctrl->channel\_stream\_info |= STREAM\_RAW1;

config\_v2\_CAMERA\_SET\_CHANNEL\_STREAM(parm1, parm2);//

该函数设置：

vfe\_interface使用：VFE0还是VFE1

设置通道和流：使用PIX还是RDI，RDI1,RDI2

ctrl->channel\_interface\_mask = ispif\_get.data.channel\_interface\_mask;

ctrl->channel\_stream\_info = ispif\_get.data.channel\_stream\_info;

使用STREAM\_IMAGE，STREAM\_RAW，STREAM\_RAW1，STREAM\_RAW2

ispif\_set.data.channel\_stream\_info = ctrl->channel\_stream\_info;

ispif\_set.data.vfe\_interface = VFE0;

config\_decide\_vfe\_outputs

config\_MSM\_V4L2\_STREAM\_ON

config\_MSG\_ID\_STOP\_ACK

mctl\_proc\_v4l2\_request桥梁

内核L:;

.vidioc\_streamon = camera\_v4l2\_streamon,

Camera.c (kernel\msm-3.18\drivers\media\platform\msm\camera\_v2\camera

内核,hal,vendor通信事件：

==

v4l2\_event\_queue

Msm.c (kernel\msm-3.18\drivers\media\platform\msm\camera\_v2)

msm\_post\_event//传送事件给上层

-===

venror如何定制它需要的内核事件？

Server.c (vendor\mm-camera\mm-camera2\server-imaging)

/\* Subcribe V4L2 event \*/

subscribe.type = MSM\_CAMERA\_V4L2\_EVENT\_TYPE;

===

hal层如何订阅内核事件：

Mm\_camera.c (hardware\camera\qcamera2\stack\mm-camera-interface\src)

mm\_camera\_evt\_sub

--->ioctl(my\_obj->ctrl\_fd, VIDIOC\_SUBSCRIBE\_EVENT, &sub);//订阅事件

监控事件：

mm\_camera\_poll\_thread\_add\_poll\_fd(&my\_obj->evt\_poll\_thread,

Streamon:

配置进程：

Qcamsvr.c (vendor\mm-camera\server\core) 36133 2019/6/20

qcamsvr\_start

Qcamsvr.c (vendor\mm-camera\server\core)

qcamsvr\_process\_server\_node\_event

Mctl.c (vendor\mm-camera\server\core\mctl)

create\_v4l2\_conf\_thread

流程：

Mctl.c (msm8937-8953\_vendor\_qcom\_proprietary\msm8937-8953\_vendor\_qcom\_proprietary\mm-camera\server\core\mctl)

---->cam\_mctl\_thread

Config\_v2.c (msm8937-8953\_vendor\_qcom\_proprietary\msm8937-8953\_vendor\_qcom\_proprietary\mm-camera\server\core\config)

mctl\_proc\_v4l2\_request

--->case MSM\_V4L2\_STREAM\_ON:

---->pme->p\_cfg\_ctrl->config\_intf->config\_request(

ctrl, ctrlCmd, &cmdPending);

Config\_v2.c (msm8937-8953\_vendor\_qcom\_proprietary\msm8937-8953\_vendor\_qcom\_proprietary\mm-camera\server\core\config)

config\_v2\_v4l2\_request

--->case MSM\_V4L2\_STREAM\_ON:

--->config\_MSM\_V4L2\_STREAM\_ON(ctrl,ctrlCmd, cmdPending);

Hal,media controller,vendor,kernel的通信：

Server.c (vendor\mm-camera\mm-camera2\server-imaging)

case RD\_FD\_HAL:

case RD\_DS\_FD\_HAL:

case RD\_PIPE\_FD\_MCT:

ISPIF\_client\_open