MSMFB\_ASYNC\_BLIT//处理图像合成

mdp3\_ctrl\_ioctl\_handler

--->case MSMFB\_ASYNC\_BLIT:

--->mdp3\_ctrl\_async\_blit\_req(mfd, argp);

--->mdp3\_ppp\_parse\_req(p\_req, &req\_list\_header, 1);

--->mdp3\_ppp\_handle\_buf\_sync(req, &req\_list\_header->sync);

--->mdp3\_ppp\_get\_img(&req->req\_list[i].src,

&req->req\_list[i], &req->src\_data[i]);

--->mdp3\_ppp\_get\_img(&req->req\_list[i].dst,

&req->req\_list[i], &req->dst\_data[i]);

--->mdp3\_ppp\_req\_push(req\_q, req);

--->schedule\_work(&ppp\_stat->blit\_work);

INIT\_WORK(&ppp\_stat->blit\_work, mdp3\_ppp\_blit\_wq\_handler);//图像合成任务

--->mdp3\_ppp\_start\_blit(mfd,&(req->req\_list[i]),&req->src\_data[i],&req->dst\_data[i]);

--->mdp3\_ppp\_blit(mfd, req, src\_data, dst\_data);

--->mdp3\_ppp\_process\_req(&blit\_op, req, src\_data, dst\_data);

--->mdp3\_start\_ppp(&blit\_op);

--->MDP3\_REG\_WRITE(0x10148, 0);

--->mdp3\_ppp\_kickoff();

--->mdp3\_irq\_enable(MDP3\_PPP\_DONE);//启动ppp中断

---->ppp\_enable();

--->ATRACE\_END("mdp3\_ppp\_start");

--->mdp3\_ppp\_req\_pop(&ppp\_stat->req\_q);

--->mdp3\_ppp\_next\_req(&ppp\_stat->req\_q);

--->&req\_q->req[req\_q->pop\_idx];

=====

Hal:

SurfaceFlinger::doComposition()

--->doDisplayComposition(hw, dirtyRegion);

--->hw->compositionComplete();

--->postFramebuffer();

--->hwc.commit();//HWComposer::commit

--->mHwc->set(mHwc, mNumDisplays, mLists);

===

通过MSMFB\_OVERLAY\_PLAY，实现了把合成后的图像数据的信息，送到驱动层，调用mdp3\_get\_img()为其在内核创建内存映射data，把data放到mdp3\_session->bufq\_in中，再通过MSMFB\_DISPLAY\_COMMIT, 把bufq\_in中的数据导出来，通过dma刷新出去，再通过mdp3\_put\_img()释放掉内存映射，完成一次图像的刷新。

--->ctx->mCopyBit[dpy]->draw(ctx, list, dpy, &fd);//下发MSMFB\_ASYNC\_BLIT，执行图像的合成

--->ctx->mHwcDebug[dpy]->dumpLayers(list);//每个layer dump下来

--->ctx->mFBUpdate[dpy]->draw(ctx, hnd)//把最后一个layer，即FRAMEBUFFER\_TRAGET的layer通过MSMFB\_OVERLAY\_PLAY刷新到kernel中去

--->Overlay::displayCommit(ctx->dpyAttr[dpy].fd, lRoi, rRoi)//调用MSMFB\_DISPLAY\_COMMIT命令，让kernel这边刷新图片

--->copybit->flush\_get\_fence(copybit, fd);  //下发MSMFB\_ASYNC\_BLIT命令通知驱动层进行图像的合成

hwc\_set\_primary()

==> ctx->mFBUpdate[dpy]->draw(ctx, hnd)

==>FBUpdateNonSplit::draw()

==> ov.queueBuffer(fd, offset, dest)

==>Overlay::queueBuffer() ==>

  mPipeBook[dest].mPipe->queueBuffer() ==>

     GenericPipe::queueBuffer() ==>

          Data::queueBuffer() ==>

             MdpData::play() ==>

                mdp\_wrapper::play() ==>

                    ioctl(fd, MSMFB\_OVERLAY\_PLAY, &od) //IOCTRL MSMFB\_OVERLAY\_PLAY

mdp3\_ctrl\_ioctl\_handler() ==> mdp3\_overlay\_play() ==> mdp3\_overlay\_queue\_buffer()

内核进程：

\_\_mdss\_fb\_display\_thread

--->\_\_mdss\_fb\_perform\_commit(mfd);

--->mdp3\_ctrl\_display\_commit\_kickoff

--->mdss\_spi\_panel\_kickoff

--->wait\_for\_completion\_timeout(&ctrl\_pdata->spi\_panel\_te,

msecs\_to\_jiffies(SPI\_PANEL\_TE\_TIMEOUT));//等待一个超时时间

--->mdss\_spi\_tx\_pixel(tx\_buf, ctrl\_pdata->byte\_pre\_frame);

Android FBIOPAN\_DISPLAY刷屏

ioctl(fbmem.c)---->

    fb\_ioctl---->

        FBIOPAN\_DISPLAY---->

             fb\_pan\_display---->

                  msm\_fb\_pan\_display---->

                       msm\_fb\_pan\_display\_ex---->

                            schedule\_work(&mfd->commit\_work)=msm\_fb\_commit\_wq\_handler---->

                                  msm\_fb\_pan\_display\_sub---->

                                        mdp\_set\_dma\_pan\_info(info, dirtyPtr,(var->activate & FB\_ACTIVATE\_VBL));

                                        mdp\_dma\_pan\_update(info);

                                        msm\_fb\_signal\_timeline(mfd);

                                        schedule\_delayed\_work(&mfd->backlight\_worker,backlight\_duration);

qcom MSMFB\_DISPLAY\_COMMIT刷屏

ioctl(fbmem.c)---->

    fb\_ioctl---->

         msm\_fb\_ioctl(mdss\_fb.c)---->

              mdss\_fb\_do\_ioctl---->

                   MSMFB\_DISPLAY\_COMMIT

                          mdss\_fb\_display\_commit---->

                              mdss\_fb\_pan\_display\_ex---->

                                    atomic\_inc(&mfd->commits\_pending);

                                    wake\_up\_all(&mfd->commit\_wait\_q);---->

                                         \_\_mdss\_fb\_display\_thread---->

                                                \_\_mdss\_fb\_perform\_commit---->

                                                        mdp3\_ctrl\_display\_commit\_kickoff---->

                                                                 mdp3\_bufq\_pop

                                                                 mdp3\_dmap\_update---->

                                                                           wait\_for\_completion\_timeout(&dma->vsync\_comp,KOFF\_TIMEOUT);

                                                                           mdp3\_vsync\_intr\_handler

                                                                           complete(&dma->vsync\_comp)

                                                                 mdss\_fb\_update\_backlight

====

gpio\_keys\_init

--->register\_call\_state\_notifier(&gpiokey\_call\_state\_notifier\_block);

--->register\_fm\_state\_notifier(&gpiokey\_call\_state\_notifier\_block);

--->return platform\_driver\_probe(&gpio\_keys\_device\_driver, gpio\_keys\_probe);

===

\*pdata = pdev->dev.platform\_data;//获取平台数据