lcd android驱动初始化流程

Lc1860平台 lcd 驱动初始化流程如下

1：获取屏的设置参数

int comipfb\_probe(void)

--->prdata = &comip\_lcd\_info;

static struct comipfb\_platform\_data comip\_lcd\_info = {

.lcdcaxi\_clk = 312000000,

.lcdc\_support\_interface = COMIPFB\_MIPI\_IF,

.phy\_ref\_freq = 26000, /\* KHz \*/

.gpio\_rst = LCD\_RESET\_PIN,

.gpio\_im = -1,

.flags = COMIPFB\_SLEEP\_POWEROFF,

};

r = &comip\_resource\_fb0[0];

struct resource comip\_resource\_fb0[] = {

[0] = {

.start = LCDC0\_BASE,

.end = LCDC0\_BASE + 0xfff,

.flags = IORESOURCE\_MEM,

},

[1] = {

.start = DSI\_BASE,

.end = MIPI\_BASE + 0xffff,

.flags = IORESOURCE\_MEM,

},

[2] = {

.start = INT\_LCDC0,

.end = INT\_LCDC0,

.flags = IORESOURCE\_IRQ,

},

};

2:lcd上电硬件初始化

display\_powerup(fbi);

-->val = readl(io\_p2v(AP\_PWR\_DISPLAY\_PD\_CTL));

val |= ((1 << AP\_PWR\_WK\_UP) | (1 << AP\_PWR\_WK\_UP\_WE));

writel(val, io\_p2v(AP\_PWR\_DISPLAY\_PD\_CTL));

while((readl(io\_p2v(AP\_PWR\_PDFSM\_ST1)) &

(0x7 << AP\_PWR\_PDFSM\_ST1\_DISPLAY\_PD\_ST)) != 0 && timeout-- > 0)

udelay(10);

3:时钟初始化使能:

clk\_init();

comipfb\_clk\_enable(fbi);

4:lcd 初始化参数设置

comipfb\_fbinfo\_init

--->

{

struct comipfb\_layer\_info \*layer;

struct comipfb\_dev \*cdev;

unsigned char i;

cdev = comipfb\_dev\_get(fbi);

if (!cdev) {

printf( "cdev is not finded !!\n");

return -EINVAL;

}

fbi->cdev = cdev;

fbi->cif = comipfb\_if\_get(fbi);

if (!fbi->cif) {

printf( "dev interface is not finded !!\n");

return -EINVAL;

}

fbi->bpp = cdev->bpp;

fbi->pixclock = cdev->pclk;

fbi->panel = 0;

fbi->refresh\_en = cdev->refresh\_en; //TODO MAYBE RGB

clk\_set\_rate(fbi->clk, fbi->pixclock);

fbi->pixclock = clk\_get\_rate(fbi->clk);

printf("fbi->pixclock = %d\n", fbi->pixclock);

clk\_enable(fbi->clk);

switch(cdev->interface\_info) {

case COMIPFB\_MIPI\_IF:

fbi->display\_mode = cdev->timing.mipi.display\_mode;

if (fbi->display\_mode == MIPI\_VIDEO\_MODE){

fbi->refresh\_en = 1;

cdev->refresh\_en = 1;

fbi->display\_info.name = cdev->name;

fbi->display\_info.xres = cdev->xres;

fbi->display\_info.yres = cdev->yres;

fbi->display\_info.pixclock = 1000000 / (fbi->pixclock / 1000000);

fbi->display\_info.sync = 0;

fbi->display\_info.left\_margin = cdev->timing.mipi.videomode\_info.hbp;

fbi->display\_info.right\_margin = cdev->timing.mipi.videomode\_info.hfp;

fbi->display\_info.upper\_margin = cdev->timing.mipi.videomode\_info.vbp;

fbi->display\_info.lower\_margin = cdev->timing.mipi.videomode\_info.vfp;

fbi->display\_info.hsync\_len = cdev->timing.mipi.videomode\_info.hsync;

fbi->display\_info.vsync\_len = cdev->timing.mipi.videomode\_info.vsync;

if (cdev->timing.mipi.videomode\_info.sync\_pol == COMIPFB\_HSYNC\_HIGH\_ACT)

fbi->display\_info.sync = FB\_SYNC\_HOR\_HIGH\_ACT;

if (cdev->timing.mipi.videomode\_info.sync\_pol == COMIPFB\_VSYNC\_HIGH\_ACT)

fbi->display\_info.sync = FB\_SYNC\_VERT\_HIGH\_ACT;

}else if (fbi->display\_mode == MIPI\_COMMAND\_MODE){

fbi->display\_info.name = cdev->name;

fbi->display\_info.xres = cdev->xres;

fbi->display\_info.yres = cdev->yres;

fbi->display\_info.pixclock = 1000000 / (fbi->pixclock / 1000000);

fbi->display\_info.left\_margin = 0;

fbi->display\_info.right\_margin = 0;

fbi->display\_info.upper\_margin = 0;

fbi->display\_info.lower\_margin = 0;

fbi->display\_info.hsync\_len = 0;

fbi->display\_info.vsync\_len = 0;

fbi->display\_info.sync = 0;

}

break;

case COMIPFB\_RGB\_IF:

break;

default:

break;

}

for (i = 0; i < ARRAY\_SIZE(comipfb\_ids); i++) {

layer = calloc(1, sizeof(struct comipfb\_layer\_info));

if (!layer)

goto failed;

layer->no = comipfb\_ids[i];

layer->parent = fbi;

if (comipfb\_layer\_init(layer, fbi) < 0)

goto failed;

fbi->layers[layer->no] = layer;

}

return 0;

failed:

comipfb\_fbinfo\_release(fbi);

return -EINVAL;

}

comipfb\_hw\_init(fbi);

--->

{

if (fbi->cdev->power)

fbi->cdev->power(fbi, 1);

else {

if (fbi->pdata->power)

fbi->pdata->power(1);

}

/\* Initialize LCD controller \*/

lcdc\_init(fbi);

return 0;

}

5:显示logo

comipfb\_show\_logo(fbi);

--->

{

struct comipfb\_layer\_info \*layer = fbi->layers[comipfb\_ids[0]];

int ret;

ret = comipfb\_read\_image(layer);

if (ret == 0) {

/\* Get logo data. \*/

comipfb\_open(layer, 0);

}

if (fbi->display\_mode == MIPI\_COMMAND\_MODE) {

lcdc\_start(fbi, 1);

}

}

6:开启背光:

comip\_backlight\_init();

{

if (CONFIG\_PWM\_EN)

backlight\_pwm\_mode\_init();

else

backlight\_gpio\_mode\_init();

}

ILI9881C屏幕的设置:

lcd\_ili9881c.c

#include "comipfb.h"

#include "comipfb\_dev.h"

#include "mipi\_cmd.h"

#include "mipi\_interface.h"

//static u8 backlight\_cmds[][ROW\_LINE] = {

// {0x00, DCS\_CMD, SW\_PACK2, 0x02, 0x51, 0xFF},

//};

static u8 lcd\_cmds\_init[][ROW\_LINE] = {

#if 1

// Start ILI9881C Initial

//{0X00, DCS\_CMD, LW\_PACK, 0x04, 0xFF, 0x98, 0x81, 0x03},

{0X00, DCS\_CMD, LW\_PACK, 0x06, 0x04, 0x00, 0xFF, 0x98, 0x81, 0x03},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x01, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x02, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x03, 0x73},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x04, 0xD3},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x05, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x06, 0x0A},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x07, 0x0E},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x08, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x09, 0x01},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x0A, 0x01},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x0B, 0x01},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x0C, 0x01},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x0D, 0x01},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x0E, 0x01},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x0F, 0x01},//

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x10, 0x01},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x11, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x12, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x13, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x14, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x15, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x16, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x17, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x18, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x19, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x1A, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x1B, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x1C, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x1D, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x1E, 0x40},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x1F, 0x80},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x20, 0x06},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x21, 0x01},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x22, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x23, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x24, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x25, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x26, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x27, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x28, 0x33},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x29, 0x03},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x2A, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x2B, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x2C, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x2D, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x2E, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x2F, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x30, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x31, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x32, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x33, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x34, 0x03},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x35, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x36, 0x03},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x37, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x38, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x39, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x3A, 0x40},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x3B, 0x40},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x3C, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x3D, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x3E, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x3F, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x40, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x41, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x42, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x43, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x44, 0x00},

//----------GIP\_2--------------

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x50, 0x01},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x51, 0x23},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x52, 0x45},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x53, 0x67},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x54, 0x89},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x55, 0xab},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x56, 0x01},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x57, 0x23},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x58, 0x45},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x59, 0x67},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x5a, 0x89},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x5b, 0xab},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x5c, 0xcd},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x5d, 0xef},

//---------GIP\_3--------------

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x5e, 0x11},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x5f, 0x08},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x60, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x61, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x62, 0x01},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x63, 0x0D},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x64, 0x0C},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x65, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x66, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x67, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x68, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x69, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x6a, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x6b, 0x0F},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x6c, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x6d, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x6e, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x6f, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x70, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x71, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x72, 0x0E},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x73, 0x06},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x74, 0x07},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x75, 0x08},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x76, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x77, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x78, 0x01},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x79, 0x0D},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x7a, 0x0C},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x7b, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x7c, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x7d, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x7e, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x7f, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x80, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x81, 0x0F},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x82, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x83, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x84, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x85, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x86, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x87, 0x02},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x88, 0x0E},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x89, 0x06},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x8a, 0x07},

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Page 4 Command \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

//{0X00, DCS\_CMD, LW\_PACK, 0X04, 0xFF, 0x98, 0x81, 0x04},

{0X00, DCS\_CMD, LW\_PACK, 0X06, 0X04, 0X00, 0xFF, 0x98, 0x81, 0x04},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x6c, 0x15},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x6e, 0x1a},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x6f, 0x33},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x8d, 0x15},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x3a, 0xa4},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x87, 0x2a},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x63, 0xC0},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x26, 0x76},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xb2, 0xd1},

//----------- Page 1 Command -------------

//{0X00, DCS\_CMD, LW\_PACK, 0x04, 0xFF, 0x98, 0x81, 0x01},

{0X00, DCS\_CMD, LW\_PACK, 0X06, 0X04, 0x00, 0xFF, 0x98, 0x81, 0x01},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x22, 0x09},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x31, 0x00},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x50, 0x9E},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x51, 0x9E},

// {0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x53, 0x5D},

// {0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x55, 0x61},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x60, 0x14},

/\*-----------------------GAMMA SETTING---------------------------------\*/

/\*-------------------------P-tive setting---------------------------\*/

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xA0, 0x00}, //255

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xA1, 0x1B}, //251

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xA2, 0x2B}, //247

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xA3, 0x15}, //243

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xA4, 0x19}, //239

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xA5, 0x2C}, //231

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xA6, 0x20}, //219

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xA7, 0x1D}, //203

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xA8, 0x98}, //175

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xA9, 0x1D}, //144

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xAA, 0x29}, //111

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xAB, 0x8A}, //80

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xAC, 0x1D}, //52

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xAD, 0x1C}, //36

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xAE, 0x50}, //24

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xAF, 0x24}, //16

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xB0, 0x2C}, //12

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xB1, 0x52}, //8

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xB2, 0x60}, //4

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xB3, 0x2A}, //0

/\*------------------------Negitive setting---------------------------\*/

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xC0, 0x00}, //255

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xC1, 0x1B}, //251

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xC2, 0x2C}, //247

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xC3, 0x16}, //243

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xC4, 0x18}, //239

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xC5, 0x2C}, //231

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xC6, 0x20}, //219

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xC7, 0x23}, //203

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xC8, 0x98}, //175

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xC9, 0x1D}, //144

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xCA, 0x2A}, //111

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xCB, 0x8A}, //80

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xCC, 0x1E}, //52

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xCD, 0x1D}, //36

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xCE, 0x50}, //24

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xCF, 0x24}, //16

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xD0, 0x2C}, //12

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xD1, 0x52}, //8

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xD2, 0x60}, //4

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0xD3, 0x2A}, //0

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Page 0 Command \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

//{0X00, DCS\_CMD, LW\_PACK, 0x04, 0xFF, 0x98, 0x81, 0x00},

{0X00, DCS\_CMD, LW\_PACK, 0X06, 0X04, 0x00, 0xFF, 0x98, 0x81, 0x00},

{0X00, DCS\_CMD, LW\_PACK, 0X03, 0x51, 0x0F, 0xFF},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x53, 0x24},

{0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x55, 0x01},

// {0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x35, 0x00},

{0X00, DCS\_CMD, SW\_PACK1, 0x01, 0x34},

// {0X00, DCS\_CMD, SW\_PACK2, 0X02, 0x36, 0x03},

//Delay\_ms(10);

//{0X78, DCS\_CMD, SW\_PACK1, 0x01, 0x11}, //modify stephen

{0X78, DCS\_CMD, SW\_PACK2, 0x02, 0x11, 0x00},

//Delay\_ms(120);

//{0X14, DCS\_CMD, SW\_PACK1, 0x01, 0x29}, //modify stephen

{0X14, DCS\_CMD, SW\_PACK2, 0x02, 0x29, 0x00},

//Delay\_ms(20);

#endif

};

static u8 lcd\_cmds\_suspend[][ROW\_LINE] = {

{0x64, DCS\_CMD, SW\_PACK1, 0x01, 0x28}, // set display off

{0x00, DCS\_CMD, SW\_PACK1, 0x01, 0x10}, // enter sleep mode

};

static u8 lcd\_cmds\_resume[][ROW\_LINE] = {

{0x78, DCS\_CMD, SW\_PACK1, 0x01, 0x11}, // exit sleep mode

{0x64, DCS\_CMD, SW\_PACK1, 0x01, 0x29}, // set display on

};

static u8 lcd\_pre\_read\_id[][ROW\_LINE] = {

//{0x00, DCS\_CMD, SW\_PACK2, 0x02, 0xFE, 0x00},

//{0x00, DCS\_CMD, LW\_PACK, 0x06, 0x04, 0x00, 0xB9, 0xFF, 0x83, 0x94},

//{0x00, DCS\_CMD, SW\_PACK2, 0x02, 0xFE, 0xF4},

//{0x00, DCS\_CMD, SW\_PACK1, 0x01, 0xFF},

//{0X00, DCS\_CMD, LW\_PACK, 0x04, 0xFF, 0x98, 0x81, 0x01},

{0x00, DCS\_CMD, LW\_PACK, 0x06, 0x06, 0x00, 0xff, 0x98, 0x81, 0x01},

{0x00, DCS\_CMD, SW\_PACK2, 0x02, 0x03, 0x00},

};

static struct common\_id\_info lcd\_common\_id\_info[] = {

//{DCS\_CMD, {0x98, 0x81, 0x00}, 3, 0x04},

{ DCS\_CMD, {0x98} , 1 , 0x00},

{ DCS\_CMD, {0x81} , 1 , 0x01},

{ DCS\_CMD, {0x0c} , 1 , 0x02},

};

static struct common\_id\_info lcd\_common\_esd\_info[] = {

};

static int lcd\_ili9881c\_power(struct comipfb\_info \*fbi, int onoff)

{

int gpio\_rst = fbi->pdata->gpio\_rst;

if (gpio\_rst < 0) {

return -ENXIO;

}

gpio\_request(gpio\_rst, "LCD Reset");

if (onoff) {

gpio\_direction\_output(gpio\_rst, 0);

pmic\_lcdio\_enable(1);

pmic\_lcdcore\_enable(1);

mdelay(50);

gpio\_direction\_output(gpio\_rst, 1);

mdelay(10);

gpio\_direction\_output(gpio\_rst, 0);

mdelay(10);

gpio\_direction\_output(gpio\_rst, 1);

mdelay(180);

} else {

gpio\_direction\_output(gpio\_rst, 0);

pmic\_lcdio\_enable(0);

pmic\_lcdcore\_enable(0);

mdelay(10);

}

gpio\_free(gpio\_rst);

return 0;

}

static int lcd\_ili9881c\_reset(struct comipfb\_info \*fbi)

{

int gpio\_rst = fbi->pdata->gpio\_rst;

if (gpio\_rst >= 0) {

gpio\_request(gpio\_rst, "LCD Reset");

gpio\_direction\_output(gpio\_rst, 1);

mdelay(10);

gpio\_direction\_output(gpio\_rst, 0);

mdelay(10);

gpio\_direction\_output(gpio\_rst, 1);

mdelay(180);

gpio\_free(gpio\_rst);

}

return 0;

}

static int lcd\_ili9881c\_suspend(struct comipfb\_info \*fbi)

{

int ret=0;

struct comipfb\_dev\_timing\_mipi \*mipi;

mipi = &(fbi->cdev->timing.mipi);

if (mipi->display\_mode == MIPI\_VIDEO\_MODE) {

mipi\_dsih\_hal\_mode\_config(fbi, 1);

}

comipfb\_if\_mipi\_dev\_cmds(fbi, &fbi->cdev->cmds\_suspend);

mdelay(20);

mipi\_dsih\_dphy\_enable\_hs\_clk(fbi, 0);

mipi\_dsih\_dphy\_reset(fbi, 0);

mipi\_dsih\_dphy\_shutdown(fbi, 0);

return ret;

}

static int lcd\_ili9881c\_resume(struct comipfb\_info \*fbi)

{

int ret=0;

struct comipfb\_dev\_timing\_mipi \*mipi;

mipi = &(fbi->cdev->timing.mipi);

mipi\_dsih\_dphy\_shutdown(fbi, 1);

mipi\_dsih\_dphy\_reset(fbi, 1);

//if (fbi->cdev->reset)

// fbi->cdev->reset(fbi);

if (!(fbi->pdata->flags & COMIPFB\_SLEEP\_POWEROFF))

ret = comipfb\_if\_mipi\_dev\_cmds(fbi, &fbi->cdev->cmds\_resume);

else

ret = comipfb\_if\_mipi\_dev\_cmds(fbi, &fbi->cdev->cmds\_init);

msleep(20);

if (mipi->display\_mode == MIPI\_VIDEO\_MODE) {

mipi\_dsih\_hal\_mode\_config(fbi, 0);

}

mipi\_dsih\_dphy\_enable\_hs\_clk(fbi, 1);

return ret;

}

struct comipfb\_dev lcd\_ili9881c\_dev = {

.name = "lcd\_ili9881c",

.interface\_info = COMIPFB\_MIPI\_IF,

.lcd\_id = LCD\_ID\_ILI9881C,

.refresh\_en = 1,

.bpp = 32,

.xres = 720,

.yres = 1280,

.flags = 0,

//.pclk = 65000000, //modify stephen

.pclk = 64000000,

.timing = {

.mipi = {

//.hs\_freq = 65000, //Kbyte modify stephen

.hs\_freq = 50000, //Kbyte

.lp\_freq = 13000, //KHZ

.no\_lanes = 4,

.display\_mode = MIPI\_VIDEO\_MODE,

.im\_pin\_val = 1,

.color\_mode = {

.color\_bits = COLOR\_CODE\_24BIT,

},

.videomode\_info = {

.hsync = 10,

.hbp = 150,

.hfp = 80,

.vsync = 10,

.vbp = 20,

.vfp = 20,

.sync\_pol = COMIPFB\_VSYNC\_HIGH\_ACT,

.lp\_cmd\_en = 1,

.lp\_hfp\_en = 1,

.lp\_hbp\_en = 1,

.lp\_vact\_en = 1,

.lp\_vfp\_en = 1,

.lp\_vbp\_en = 1,

.lp\_vsa\_en = 1,

.mipi\_trans\_type = VIDEO\_NON\_BURST\_WITH\_SYNC\_PULSES,

},

.phytime\_info = {

.clk\_tprepare = 3, //HSBYTECLK

},

.teinfo = {

.te\_source = 1, //external signal

.te\_trigger\_mode = 0,

.te\_en = 0,

.te\_sync\_en = 0,

},

.ext\_info = {

.eotp\_tx\_en = 0,

},

},

},

.panel\_id\_info = {

.id\_info = lcd\_common\_id\_info,

.num\_id\_info = 3,

.prepare\_cmd = {ARRAY\_AND\_SIZE(lcd\_pre\_read\_id)},

},

.cmds\_init = {ARRAY\_AND\_SIZE(lcd\_cmds\_init)},

.cmds\_suspend = {ARRAY\_AND\_SIZE(lcd\_cmds\_suspend)},

.cmds\_resume = {ARRAY\_AND\_SIZE(lcd\_cmds\_resume)},

.power = lcd\_ili9881c\_power,

.reset = lcd\_ili9881c\_reset,

.suspend = lcd\_ili9881c\_suspend,

.resume = lcd\_ili9881c\_resume,

// .backlight\_info = {ARRAY\_AND\_SIZE(backlight\_cmds),

// .brightness\_bit = 8,

// },

#ifdef CONFIG\_FB\_COMIP\_ESD

.esd\_id\_info = {

.id\_info = lcd\_common\_esd\_info,

.num\_id\_info = 1,

},

#endif

};

int lcd\_ili9881c\_init(void)

{

return comipfb\_dev\_register(&lcd\_ili9881c\_dev);

}