

# A64

## LCD 使用说明书

Confidential

# 文档履历

版本号	日期	制/修订人	制/修订记录
V1.0	2015-06-3		建立初始版本

Confidential

# 目 录

<b>1. 概述 .....</b>	<b>4</b>
1.1. 编写目的.....	4
1.2. 适用范围.....	4
1.3. 相关人员.....	4
<b>2. 相关配置 .....</b>	<b>5</b>
2.1. menuconfig 配置说明.....	5
2.2. sys_config 配置说明 .....	6
<b>3. 屏驱动源码位置.....</b>	<b>9</b>
3.1. 源码位置.....	9
<b>4. 新屏驱动支持说明.....</b>	<b>10</b>
<b>5. 硬件参数说明 .....</b>	<b>11</b>
5.1. LCD 接口参数说明.....	11
5.1.1. lcd_if.....	11
5.1.2. lcd_hv_if.....	11
5.1.3. lcd_hv_clk_phase.....	11
5.1.4. lcd_hv_sync_polarity.....	11
5.1.5. lcd_hv_srgb_seq.....	12
5.1.6. lcd_hv_syuv_seq.....	12
5.1.7. lcd_hv_syuv_fdly.....	12
5.1.8. lcd_cpu_if.....	12
5.1.9. lcd_cpu_te.....	13
5.1.10. lcd_lvds_if.....	13
5.1.11. lcd_lvds_colordepth.....	13
5.1.12. lcd_lvds_mode.....	13
5.1.13. lcd_dsi_if.....	14
5.1.14. lcd_dsi_lane.....	14
5.1.15. lcd_dsi_format.....	15
5.1.16. lcd_dsi_te.....	15
5.2. LCD 时序参数说明.....	16
5.2.1. lcd_x.....	16
5.2.2. lcd_y.....	16
5.2.3. lcd_ht.....	16
5.2.4. lcd_hbp.....	16
5.2.5. lcd_hspw.....	16
5.2.6. lcd_vt.....	16
5.2.7. lcd_vbp.....	16
5.2.8. lcd_vspw.....	16
5.2.9. lcd_dclk_freq.....	17
5.3. LCD 其他参数说明.....	18

5.3.1. lcd_width.....	18
5.3.2. lcd_height.....	18
5.3.3. lcd_pwm_used .....	18
5.3.4. lcd_pwm_ch.....	18
5.3.5. lcd_pwm_freq.....	18
5.3.6. lcd_pwm_pol.....	18
5.3.7. lcd_pwm_pol.....	18
5.3.8. lcd_pwm_max_limit .....	18
5.3.9. lcd_frm.....	18
5.3.10. lcd_gamma_en.....	19
5.3.11. Lcd_bl_n_percent.....	19
5.3.12. lcd_cmap_en.....	20
5.4. POWER 及 IO 说明.....	21
5.4.1. lcd_power .....	21
5.4.2. lcd_bl_en.....	21
5.4.3. lcd_bl_regulator.....	21
5.4.4. lcd_gpio_x.....	21
5.4.5. lcddx.....	21
5.4.6. lcd_io_regulator.....	22
<b>6. 屏驱动说明.....</b>	<b>23</b>
6.1. 屏驱动说明 .....	23
6.2. 开关屏流程.....	23
6.2.1. 开关屏流程函数说明 .....	24
6.2.2. 屏驱动可使用接口说明 .....	25
6.3. 屏的初始化.....	26
6.3.1. MIPI DSI 屏的初始化.....	26
6.3.2. CPU/I80 屏的初始化 .....	26
6.3.3. 使用 IO 模拟串行接口初始化.....	27
6.3.4. 使用 iic/spi 串行接口初始化.....	28

## 1. 概述

### 1.1. 编写目的

介绍 A64 项目 lcd 使用方法。

### 1.2. 适用范围

A64 平台

### 1.3. 相关人员

系统整合人员，显示开发相关人员，客户

Confidential



## 2.2. sys\_config 配置说明

sys\_config 模板如下所示:

```

;-----
;disp init configuration
;
;disp_mode          (0:screen0<screen0,fb0>)
;screenx_output_type (0:none; 1:lcd; 3:hdmi;)
;screenx_output_mode (used for hdmi output, 0:480i 1:576i 2:480p 3:576p 4:720p50)
;                   (5:720p60 6:1080i50 7:1080i60 8:1080p24 9:1080p50 10:1080p60)
;fbx format         (0:ARGB 1:ABGR 2:RGBA 3:BGRA 5:RGB565 8:RGB888 12:ARGB4444
16:ARGB1555 18:RGBA5551)
;fbx_width,fbx_height (framebuffer horizontal/vertical pixels, fix to output resolution while equal 0)
;-----
[disp]
disp_init_enable    = 1
disp_mode           = 0

screen0_output_type = 1
screen0_output_mode = 4

screen1_output_type = 3
screen1_output_mode = 4

fb0_format          = 0
fb0_width           = 0
fb0_height          = 0

fb1_format          = 0
fb1_width           = 0
fb1_height          = 0
;-----
;lcd0 configuration

;lcd_if:            0:lv(sync+de); 1:8080; 2:ttl; 3:lvds; 4:dsi; 5:edp
;lcd_backlight     lcd init backlight
;lcd_x:            lcd horizontal resolution
;lcd_y:            lcd vertical resolution
;lcd_width:        width of lcd in mm
;lcd_height:       height of lcd in mm
;lcd_dclk_freq:    in MHZ unit
;lcd_pwm_freq:     in HZ unit
;lcd_pwm_pol:      lcd backlight PWM polarity
;lcd_pwm_max_limit lcd backlight PWM max limit(<=255)

```

```

;lcd_hbp:          hsync back porch
;lcd_ht:           hsync total cycle
;lcd_vbp:          vsync back porch
;lcd_vt:           vsync total cycle
;lcd_hspw:         hsync plus width
;lcd_vspw:         vsync plus width
;lcd_lvds_if:      0:single link; 1:dual link
;lcd_lvds_colordepth: 0:8bit; 1:6bit
;lcd_lvds_mode:    0:NS mode; 1:JEIDA mode
;lcd_frm:          0:disable; 1:enable rgb666 dither; 2:enable rgb656 dither
;lcd_hv_clk_phase  lcd hv panel lock phase, 0:0 degree; 1:90 degree; 2: 180 degree; 3: 270 degree
;lcd_hv_sync_polarity lcd hv panel sync signals polarity
;                  0:vsync active low, hsync active low; 1:vsync active high, hsync active low
;                  2:vsync active low, hsync active high; 3:vsync active high, hsync active high
;lcd_gamma_en      lcd gamma correction enable
;lcd_bright_curve_en lcd bright curve correction enable
;lcd_cmap_en       lcd color map function enable
;-----
[lcd0]
lcd_used           = 1

lcd_driver_name    = "dx0960be40a1"
lcd_backlight      = 50
lcd_if             = 4
lcd_x              = 800
lcd_y              = 1280
lcd_width          = 129
lcd_height         = 206
lcd_dclk_freq      = 103
lcd_pwm_used       = 1
lcd_pwm_ch         = 0
lcd_pwm_freq       = 50000
lcd_pwm_pol        = 1
lcd_pwm_max_limit  = 250
lcd_hbp           = 60
lcd_ht            = 1300
lcd_hspw          = 20
lcd_vbp           = 21
lcd_vt            = 1332
lcd_vspw          = 10
lcd_lvds_if        = 0
lcd_lvds_colordepth = 0
lcd_lvds_mode      = 0
lcd_frm           = 0

```



```
lcd_hv_clk_phase    = 0
lcd_hv_sync_polarity= 0
lcd_dsi_if          = 0
lcd_dsi_lane        = 4
lcd_dsi_format      = 0
lcd_dsi_te          = 0
lcd_gamma_en        = 0
lcd_bright_curve_en= 0
lcd_cmap_en         = 0

lcd_bl_en           = port:PD23<1><0><default><1>
lcd_bl_en_power     = "none"
lcd_power           = "vcc-mipi"
lcd_fix_power       = "vcc-dsi-33"
;-----
;pwm config
;-----
[pwm]
pwm_used            = 1
pwm_positive        = port:PD22<2><0><default><default>
```

### 3. 屏驱动源码位置

#### 3.1. 源码位置

linux3-10/drivers/video/sunxi/disp2/disp/lcd/

sonfidentia

## 4. 新屏驱动支持说明

1. 新添加屏驱动，在源码目录下添加新的屏驱动，以 `default_panel.c` 为模板，新建立一个屏驱动，修改开关屏的流程，以及屏驱动的名字，名字须与 `sys_config` 中的 `lcd_driver_name` 对应。
2. 在 `panels.c` 中，将新的屏驱动加到 `panel_array` 中。
3. 修改 Makefile 文件
4. 在 `sys_config` 中，配置屏参数，`lcd0`，各参数见后续详细说明。
5. 在 `sys_config` 中，配置屏的相关 pin 脚，`power`，`pwm` 等。
6. 编译，加载 `disp.ko`，屏亮!!!

Confidential

## 5. 硬件参数说明

### 5.1. LCD 接口参数说明

#### 5.1.1. lcd\_if

Lcd Interface

设置相应值的对应含义为:

- 0: HV RGB 接口
- 1: CPU/I80 接口
- 2: Reserved
- 3: LVDS 接口
- 4: DSI 接口
- 5: eDP 接口

#### 5.1.2. lcd\_hv\_if

Lcd HV panel Interface

这个参数只有在 lcd\_if=0 时才有效。定义 RGB 同步屏下的几种接口类型。

设置相应值的对应含义为:

- 0: Parallel RGB
- 8: Serial RGB
- 10: Dummy RGB
- 11: RGB Dummy
- 12: Serial YUV (CCIR656)

#### 5.1.3. lcd\_hv\_clk\_phase

Lcd HV panel Clock Phase

这个参数只有在 lcd\_if=0 时才有效。定义 RGB 同步屏的 clock 与 data 之间的相位关系。

总共有 4 个相位可供调节。

设置相应值的对应含义为:

- 0: 0°
- 1: 90°
- 2: 180°
- 3: 270°

#### 5.1.4. lcd\_hv\_sync\_polarity

Lcd HV panel Sync signals Polarity

这个参数只有在 lcd\_if=0 时才有效。定义 RGB 同步屏的 hsync 和 vsync 的极性。

设置相应值的对应含义为:

- 0: vsync active low, hsync active low
- 1: vsync active high, hsync active low

2: vsync active low, hsync active high

3: vsync active high, hsync active high

### 5.1.5. `lcd_hv_srgb_seq`

Lcd HV panel Serial RGB output Sequence

这个参数只有在 `lcd_if=0` 且 `lcd_hv_if=1` (Serial RGB) 时才有效。

定义奇数行 RGB 输出的顺序:

0: Odd lines R→G→B; Even line R→G→B

1: Odd lines B→R→G; Even line R→G→B

2: Odd lines G→B→R; Even line R→G→B

4: Odd lines R→G→B; Even line B→R→G

5: Odd lines B→R→G; Even line B→R→G

6: Odd lines G→B→R; Even line B→R→G

8: Odd lines R→G→B; Even line G→B→R

9: Odd lines B→R→G; Even line G→B→R

10: Odd lines G→B→R; Even line G→B→R

### 5.1.6. `lcd_hv_syuv_seq`

Lcd HV panel Serial YUV output Sequence

这个参数只有在 `lcd_if=0` 且 `lcd_hv_if=2` (Serial YUV) 时才有效。

定义 YUV 输出格式:

0: YUYV

1: YVYU

2: UYVY

3: VYUY

### 5.1.7. `lcd_hv_syuv_fdly`

Lcd HV panel Serial YUV F line Delay

这个参数只有在 `lcd_if=0` 且 `lcd_hv_if=2` (Serial YUV) 时才有效。

定义 CCIR656 编码时 F 相对有效行延迟的行数:

0: F toggle right after active video line

1: Delay 2 lines (CCIR NTSC)

2: Delay 3 lines (CCIR PAL)

### 5.1.8. `lcd_cpu_if`

Lcd CPU panel Interface

这个参数只有在 `lcd_if=1` 时才有效。

设置相应值的对应含义为:

0: 18bit/1cycle parallel (RGB666)

4: 16bit/1cycle parallel (RGB565)

6: 18bit/3cycle parallel (RGB666)

7: 16bit/2cycle parallel (RGB565)

### 5.1.9. **lcd\_cpu\_te**

Lcd CPU panel tear effect

设置相应值的对应含义为，设置为 0 时，刷屏间隔时间为  $lcd\_ht \times lcd\_vt$ ；设置为 1 或 2 时，刷屏间隔时间为两个 te 脉冲：

- 0: frame triggered automatically
- 1: frame triggered by te rising edge
- 2: frame triggered by te falling edge

### 5.1.10. **lcd\_lvds\_if**

Lcd LVDS panel Interface

设置相应值的对应含义为：

- 0: Single Link( 1 clock pair+3/4 data pair)
- 1: Dual Link(2 clock pair)

### 5.1.11. **lcd\_lvds\_colordepth**

Lcd LVDS panel color depth

设置相应值对应含义为：

- 0: 8bit per color(4 data pair)
- 1: 6bit per color(3 data pair)

### 5.1.12. **lcd\_lvds\_mode**

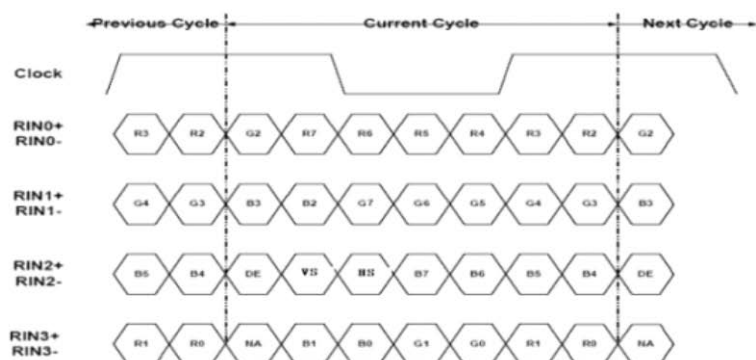
Lcd LVDS Mode

这个参数只有在  $lcd\_lvds\_bitwidth=0$  时才有效

设置相应值对应含义为(见下图)：

- 0: NS mode
- 1: JEIDA mode

### JEDIA mode



### NS mode

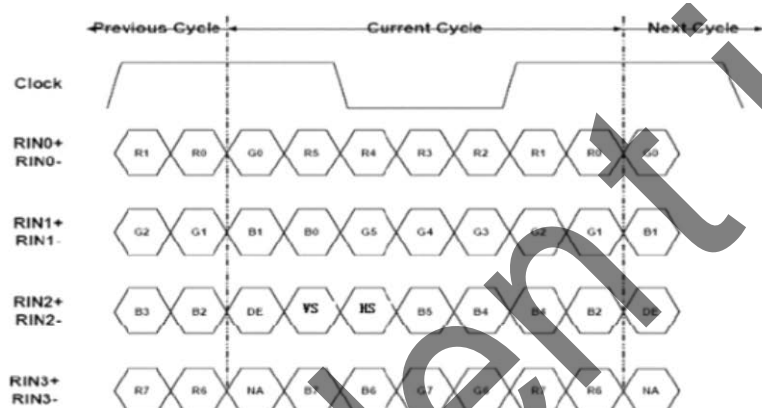


图 5-1 LVDS JEDIA mode and NS mode

#### 5.1.13. `lcd_dsi_if`

Lcd MIPI DSI panel Interface

这个参数只有在 `lcd_if=4` 时才有效。定义 MIPI DSI 屏的两种类型。

设置相应值的对应含义为：

- 0: Video mode
- 1: Command mode
- 2: video burst mode

注：Video mode 的 LCD 屏，是实时刷屏的，有 `ht`, `hbp` 等时序参数的定义；Command mode 的屏，屏上带有显示 Buffer，一般会有一个 TE 引脚

#### 5.1.14. `lcd_dsi_lane`

Lcd MIPI DSI panel Data Lane number

这个参数只有在 `lcd_if=4` 时才有效。

设置相应值的对应含义为：

- 1: 1 data lane

- 2: 2 data lane
- 3: 3 data lane
- 4: 4 data lane

#### 5.1.15. `lcd_dsi_format`

Lcd MIPI DSI panel Data Pixel Format

这个参数只有在 `lcd_if=4` 时才有效。

设置相应值的对应含义为:

- 0: Package Pixel Stream, 24bit RGB
- 1: Loosely Package Pixel Stream, 18bit RGB
- 2: Package Pixel Stream, 18bit RGB
- 3: Package Pixel Stream, 16bit RGB

#### 5.1.16. `lcd_dsi_te`

Lcd MIPI DSI panel Tear Effect

这个参数只有在 `lcd_if=4` 时才有效。

设置相应值的对应含义为:

- 0: frame triggered automatically
- 1: frame triggered by te rising edge
- 2: frame triggered by te falling edge

注: 设置为 0 时, 刷屏间隔时间为  $lcd\_ht \times lcd\_vt$ ; 设置为 1 或 2 时, 刷屏间隔时间为两个 `te` 脉冲



## 5.2. LCD 时序参数说明

### 5.2.1. lcd\_x

显示屏的水平像素点

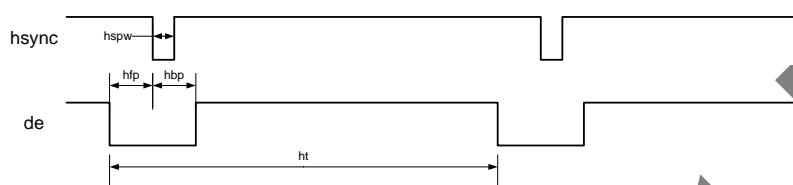
### 5.2.2. lcd\_y

显示屏的垂直像素点

### 5.2.3. lcd\_ht

Horizontal Total time

指一行总的 dclk 的 cycle 个数。见下图：



### 5.2.4. lcd\_hbp

Horizontal Back Porch

指有效行间，行同步信号 (hsync) 开始，到有效数据开始之间的 dclk 的 cycle 个数，包括同步信号区。见上图，注意的是包含了 hspw 段。

### 5.2.5. lcd\_hspw

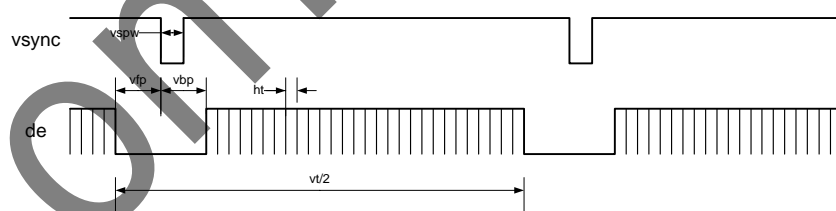
Horizontal Sync Pulse Width

指行同步信号的宽度。单位为 1 个 dclk 的时间 (即是 1 个 data cycle 的时间)。见上图。

### 5.2.6. lcd\_vt

Vertical Total time

指一场的总行数。见下图：



### 5.2.7. lcd\_vbp

Vertical Back Porch

指场同步信号 (vsync) 开始，到有效数据行开始之间的行数，包括场同步信号区。见上图，注意的是包含了 vspw 段。

### 5.2.8. lcd\_vspw

#### Vertical Sync Pulse Width

指场同步信号的宽度。单位为行。见上图。

#### 5.2.9. lcd\_dclk\_freq

##### Data Clock Frequency

指 PIN 总线上数据的传送频率。单位为 MHz

屏幕刷新帧数 =  $(\text{lcd\_dclk\_freq} \times 1000 \times 1000) / (\text{ht} \times \text{vt})$ 。

Confidential

### 5.3. LCD 其他参数说明

#### 5.3.1. lcd\_width

Width of lcd panel in mm

此参数描述 lcd 屏幕的物理宽度，单位是 mm。用于计算 dpi

#### 5.3.2. lcd\_height

height of lcd panel in mm

此参数描述 lcd 屏幕的物理高度，单位是 mm。用于计算 dpi

#### 5.3.3. lcd\_pwm\_used

If used pwm

此参数标识是否使用 pwm 用以背光亮度的控制。

#### 5.3.4. lcd\_pwm\_ch

Pwm channel used

此参数标识使用的 Pwm 通道。

#### 5.3.5. lcd\_pwm\_freq

Lcd backlight PWM Frequency

这个参数配置 PWM 信号的频率，单位为 Hz。

#### 5.3.6. lcd\_pwm\_pol

Lcd backlight PWM Polarity

这个参数配置 PWM 信号的占空比的极性。设置相应值对应含义为：

0: active high

1: active low

#### 5.3.7. lcd\_pwm\_pol

Lcd backlight PWM Polarity

这个参数配置 PWM 信号的占空比的极性。设置相应值对应含义为：

0: active high

1: active low

#### 5.3.8. lcd\_pwm\_max\_limit

Lcd backlight PWM 最高限制，以亮度值表示

比如 150，则表示背光最高只能调到 150，0~255 范围内的亮度值将会被线性映射到 0~150 范围内。用于控制最高背光亮度，节省功耗。

#### 5.3.9. lcd\_frm

### Lcd Frame Rate Modulator

FRM 是解决由于 PIN 减少导致的色深问题。

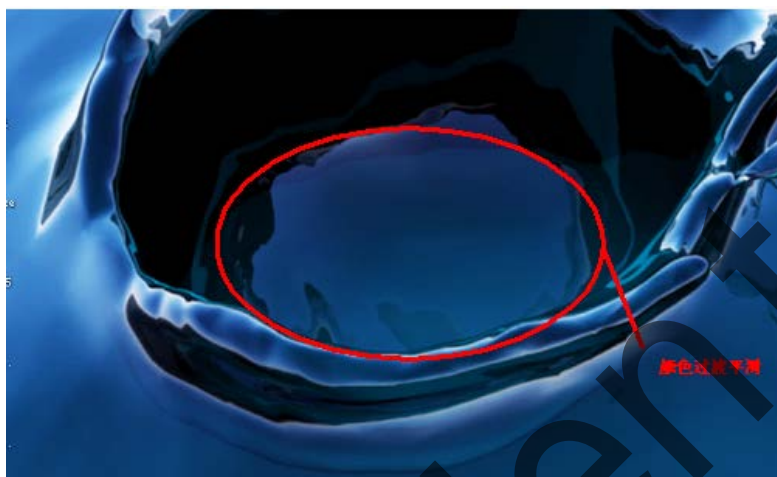
这个参数设置相应值对应含义为：

0: RGB888 → RGB888 direct

1: RGB888 → RGB666 dither

2: RGB888 → RGB565 dither

有些 LCD 屏的像素格式是 18bit 色深 (RGB666) 或 16bit 色深 (RGB565)，建议打开 FRM 功能，通过 dither 的方式弥补色深，使显示达到 24bit 色深 (RGB888) 的效果。如下图所示，上图是色深为 RGB66 的 LCD 屏显示，下图是打开 dither 后的显示，打开 dither 后色彩渐变的地方过度平滑。



### 5.3.10. lcd\_gamma\_en

Lcd Gamma Correction Enable

设置相应值的对应含义为：

0: Lcd 的 Gamma 校正功能关闭

1: Lcd 的 Gamma 校正功能开启

设置为 1 时，需要在屏驱动中对 lcd\_gamma\_tbl [256] 进行赋值。

### 5.3.11. Lcd\_bl\_n\_percent

背光映射值, n 为(0-100)

此功能是针对亮度非线性的 LCD 屏的, 按照配置的亮度曲线方式来调整亮度变化, 以使亮度变化更线性。

比如 lcd\_bl\_50\_percent = 60, 表明将 50% 的亮度值调整成 60%, 即亮度比原来提高 10%。

### 5.3.12. lcd\_cmap\_en

Lcd Color Map Enable

设置相应值的对应含义为:

0: Lcd 的色彩映射功能关闭

1: Lcd 的色彩映射功能开启

设置为 1 时, 需要对 lcd\_cmap\_tbl [2][3][4]进行赋值 Lcd Color Map Table。

每个像素有 R、G、B 三个单元, 每四个像素组成一个选择项, 总共有 12 个可选。数组第一维表示奇偶行, 第二维表示像素的 RGB, 第三维表示第几个像素, 数组的内容即表示该位置映射到的内容。

LCD CMAP 是对像素的映射输出功能, 只有像素有特殊排布的 LCD 屏才需要配置。

LCD CMAP 定义每行的 4 个像素为一个总单元, 每个像素分 R、G、B 3 个小单元, 总共有 12 个小单元。通过 lcd\_cmap\_tbl 定义映射关系, 输出的每个小单元可随意映射到 12 个小单元之一。

```
__u32 lcd_cmap_tbl[2][3][4] = {
    {
        {LCD_CMAP_G0,LCD_CMAP_B1,LCD_CMAP_G2,LCD_CMAP_B3},
        {LCD_CMAP_B0,LCD_CMAP_R1,LCD_CMAP_B2,LCD_CMAP_R3},
        {LCD_CMAP_R0,LCD_CMAP_G1,LCD_CMAP_R2,LCD_CMAP_G3},
    },
    {
        {LCD_CMAP_B3,LCD_CMAP_G2,LCD_CMAP_B1,LCD_CMAP_G0},
        {LCD_CMAP_R3,LCD_CMAP_B2,LCD_CMAP_R1,LCD_CMAP_B0},
        {LCD_CMAP_G3,LCD_CMAP_R2,LCD_CMAP_G1,LCD_CMAP_R0},
    },
};
```

如上, 上三行代表奇数行的像素排布, 下三行代表偶数行的像素排布;

每四个像素为一个单元, 第一列代表每四个像素的第一个像素映射, 第二列代表每四个像素的第二个像素映射, 以此类推。

如上的定义, 像素的输出格式如下图所示。

奇数行	G0	B1	G2	B3	G4	B5	G6	B7
	B0	R1	G2	R3	B4	R5	G6	R7
	R0	G1	R2	G3	R4	G5	R6	G7
偶数行	B3	G2	B1	G0	B7	G6	B5	G4
	R3	G2	R1	B0	R7	G6	R5	B4
	G3	R2	G1	R0	G7	R6	G5	R4

## 5.4. POWER 及 IO 说明

### 5.4.1. lcd\_power

示例: `lcd_power = "vcc-lcd"`

配置 regulator 的名字。配置好之后, 需要在屏驱动调用相应的接口进行开、关的控制。  
Power 的配置时建议根据 axp 使用文档进行。

### 5.4.2. lcd\_bl\_en

示例: `lcd_bl_en = port:PD24<1><0><default><1>`

含义: `lcd_power` 引脚为 PD24, PD24 输出高电平时打开 LCD 背光; 上下拉不使能。

第一个尖括号: 功能分配; 1 为输出;

第二个尖括号: 内置电阻; 使用 0 的话, 标示内部电阻高阻态, 如果是 1 则是内部电阻上拉, 2 就代表内部电阻下拉。使用 `default` 的话代表默认状态, 即电阻上拉。其它数据无效。

第三个尖括号: 驱动能力; `default` 表驱动能力是等级 1

第四个尖括号: 输出有效所需电平; LCD 背光工作时的电平, 0 为低电平, 1 为高电平。

需要在屏驱动调用相应的接口进行开、关的控制。

### 5.4.3. lcd\_bl\_regulator

示例: `lcd_bl_regulator = "xxx"`

配置 `lcd_bl_en` 的 io 的电源。这电源会在 `lcd_bl_en` 使能时自动开启。

### 5.4.4. lcd\_gpio\_x

示例: `lcd_gpio_0 = port:PD25<0><0><default><0>`

含义: `lcd_gpio_0` 引脚为 PD25。

第一个尖括号: 功能分配; 0 为输入, 1 为输出;

第二个尖括号: 内置电阻; 使用 0 的话, 标示内部电阻高阻态, 如果是 1 则是内部电阻上拉, 2 就代表内部电阻下拉。使用 `default` 的话代表默认状态, 即电阻上拉。其它数据无效。

第三个尖括号: 驱动能力; `default` 表驱动能力是等级 1

第四个尖括号: 表示默认值; 即是当设置为输出时, 该引脚输出的电平, 0 为低电平, 1 为高电平。

需要在屏驱动调用相应的接口进行拉高, 拉低的控制。

### 5.4.5. lcd\_dx

示例: `lcd_dx0 = port:PD00<3><0><default><default>`

含义: `lcd_dx0` 这个引脚, 即是 PD0, 配置为 LVDS 输出。

第一个尖括号: 功能分配; 0 为输入, 1 为输出, 2 为 LCD 输出, 3 为 LVDS 接口输出, 7 为 `disable`。

第二个尖括号: 内置电阻; 使用 0 的话, 标示内部电阻高阻态, 如果是 1 则是内部电阻上拉, 2 就代表内部电阻下拉。使用 `default` 的话代表默认状态, 即电阻上拉。其它数据无效。

第三个尖括号: 驱动能力; `default` 表驱动能力是等级 1

第四个尖括号：表示默认值；即是当设置为输出时，该引脚输出的电平，0 为低电平，1 为高电平。

LCD PIN 的配置如下：

LCD 为 HV RGB 屏，CPU/I80 屏时，必须定义相应的 IO 口为 LCD 输出（如果是 0 路输出，第一个尖括号为 2；如果是 1 路输出，第一个尖括号为 3）；

具体的 IO 对应关系可参考 user manual 手册进行配置。

LCD PIN 的所有 IO，均可通过注释方式去掉其定义，显示驱动对注释 IO 不进行初始化操作。

需要在屏驱动调用相应的接口进行开、关的控制。

#### 5.4.6. lcd\_io\_regulator

示例：`lcd_io_regulator = "vcc-pd"`

配置 lcd 的电源。这电源会在 lcd 的 io 使能时自动开启。

## 6. 屏驱动说明

### 6.1. 屏驱动说明

屏驱动中需要实现的函数接口有 LCD\_cfg\_panel\_info, LCD\_open\_flow, LCD\_close\_flow 和 LCD\_get\_panel\_funs\_0/LCD\_get\_panel\_funs\_1 是必须包含的 4 个函数。

#### 函数: LCD\_cfg\_panel\_info

功能: 配置的 TCON 扩展参数

原型:

```
static void LCD_cfg_panel_info(__panel_extend_para_t * info)
```

TCON 的扩展参数只能在屏文件中配置, 参数的定义见“5.3 LCD 其他参数说明”。

需要 gamma 校正, 或色彩映射, 在 sys\_config 中将相应模块的 enable 参数置 1, lcd\_gamma\_en, lcd\_cmap\_en, 并且填充 3 个系数表, lcd\_gamma\_tbl, lcd\_cmap\_tbl, 如下所示红色代码部分。注意的是: gamma, 模板提供了 18 段拐点值, 然后再插值出所有的值 (255 个)。如果觉得还不细, 可以往相应表格里添加子项。cmap\_tbl 的大小是固定了, 不能减小或增加表的大小。

最终生成的 gamma 表项是由 rgb 三个 gamma 值组成的, 各占 8bit, 目前提供的模板中, 三个 gamma 值是相同的。

#### 函数: LCD\_open\_flow

功能: 定义开屏的流程

原型: static \_\_s32 LCD\_open\_flow(\_\_u32 sel)

具体说明见“6.2 开关屏流程”。

#### 函数: LCD\_close\_flow

功能: 定义关屏的流程

原型: static \_\_s32 LCD\_close\_flow(\_\_u32 sel)

该函数与 LCD\_open\_flow 对应

#### 屏驱动: \_\_lcd\_panel\_t default\_panel

功能:

原型: 名字可改成屏相对应的名字。

### 6.2. 开关屏流程

开关屏的操作流程如图 6-2 所示。

其中, LCD\_open\_flow 和 LCD\_close\_flow 称为开关屏流程函数, 方框中的函数, 如 LCD\_power\_on, TCON\_open 等函数, 称为开关屏步骤函数。

不需要进行初始化操作的 LCD 屏, LCD\_panel\_init 及 LCD\_panel\_exit 这函数可以为空。



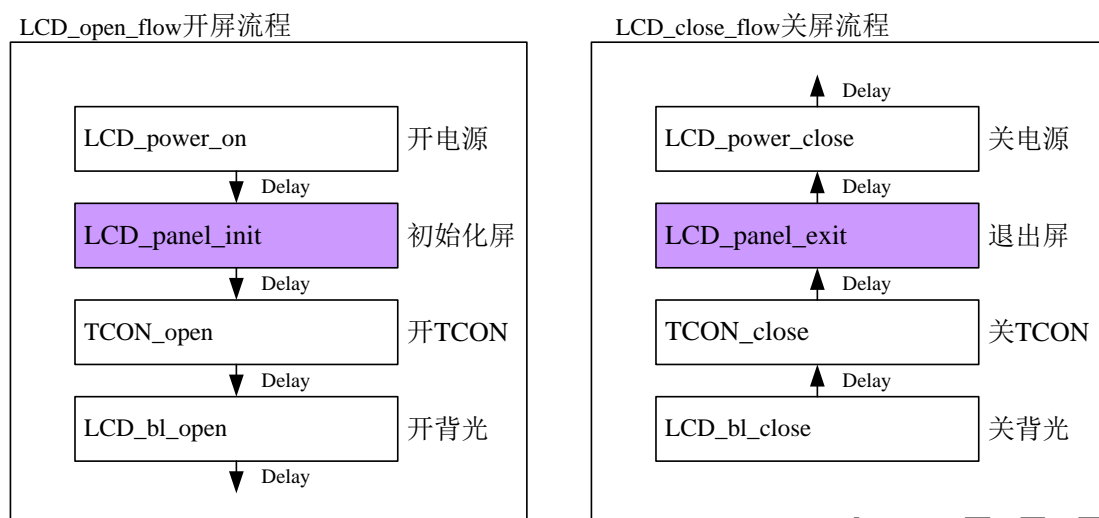


图 6-1 开关屏流程

### 6.2.1. 开关屏流程函数说明

#### 函数: LCD\_open\_flow

功能: 初始化开关屏的步骤流程

原型: static \_\_s32 LCD\_open\_flow(\_\_u32 sel)

函数常用内容为:

```
static __s32 LCD_open_flow(__u32 sel)
{
    LCD_OPEN_FUNC(sel, LCD_power_on, 10);
    LCD_OPEN_FUNC(sel, LCD_panel_init, 50);
    LCD_OPEN_FUNC(sel, sunxi_lcd_tcon_enable, 100);
    LCD_OPEN_FUNC(sel, LCD_bl_open, 0);
    return 0;
}
```

如上, 初始化整个开屏的流程步骤为四个:

打开 LCD 电源, 再延迟 10ms;

初始化屏, 再延迟 50ms; (不需要初始化的屏, 可省掉此步骤)

打开 TCON, 再延迟 100ms;

打开背光, 再延迟 0ms。

LCD\_open\_flow 函数只会系统初始化的时候调用一次, 执行每个 LCD\_OPEN\_FUNC 即是把对应的开屏步骤函数进行注册, 并没有执行该开屏步骤函数。LCD\_open\_flow 函数的内容必须统一用 LCD\_OPEN\_FUNC(sel, function, delay\_time) 进行函数注册的形式, 确保正常注册到开屏步骤中。

#### 函数: LCD\_OPEN\_FUNC

功能: 注册开屏步骤函数到开屏流程中

原型: void LCD\_OPEN\_FUNC(\_\_u32 sel, LCD\_FUNC func, \_\_u32 delay)

参数说明:

func 是一个函数指针, 其类型是: void (\*LCD\_FUNC) (\_\_u32 sel), 用户自己定义的函

数必须也要用统一的形式。比如：

```
void user_defined_func(__u32 sel)
{
    //do something
}
```

delay 是执行该步骤后，再延迟的时间，时间单位是毫秒。

### 6.2.2. 屏驱动可使用接口说明

**函数：sunxi\_lcd\_delay\_ms/sunxi\_lcd\_delay\_us**

功能：延时函数，分别是毫秒级别/微秒级别的延时

原型：s32 sunxi\_lcd\_delay\_ms(u32 ms); / s32 sunxi\_lcd\_delay\_us(u32 us);

**函数：sunxi\_lcd\_tcon\_enable /sunxi\_lcd\_tcon\_disable**

功能：打开 LCD 控制器，开始刷新 LCD 显示。关闭 LCD 控制器，停止刷新数据。

原型：void sunxi\_lcd\_tcon\_enable(u32 screen\_id);/ void sunxi\_lcd\_tcon\_disable(u32 screen\_id);

**函数：sunxi\_lcd\_backlight\_enable/ sunxi\_lcd\_backlight\_disable**

功能：打开/关闭背光，操作的是 sys\_config 中 lcd\_bl 配置的 gpio。见 5.4.2 lcd\_bl\_en

原型：void sunxi\_lcd\_backlight\_enable(u32 screen\_id);

void sunxi\_lcd\_backlight\_disable(u32 screen\_id);

**函数：sunxi\_lcd\_pwm\_enable / sunxi\_lcd\_pwm\_disable**

功能：打开/关闭 pwm 控制器，打开时 pwm 将往外输出 pwm 波形。对应的是 lcd\_pwm\_ch 所对应的那一路 pwm

原型：s32 sunxi\_lcd\_pwm\_enable(u32 screen\_id);

s32 sunxi\_lcd\_pwm\_disable(u32 screen\_id);

**函数：sunxi\_lcd\_power\_enable / sunxi\_lcd\_power\_disable**

功能：打开/关闭 Lcd 电源，操作的是 sys\_config 中的 lcd\_power/lcd\_power1/lcd\_power2。

(pwr\_id 标识电源索引)

原型：void sunxi\_lcd\_power\_enable(u32 screen\_id, u32 pwr\_id);

void sunxi\_lcd\_power\_disable(u32 screen\_id, u32 pwr\_id);

**函数：sunxi\_lcd\_pin\_cfg**

功能：配置 lcd 的 io。

原型：s32 sunxi\_lcd\_pin\_cfg(u32 screen\_id, u32 bon);

说明：配置 lcd 的 data/clock 等 pin，对应 sys\_config 中的 lcd0-lcd23/lcdclk/lcdde/lcdhsync/lcdvsync。

由于 dsi 是专用 pin,所以 dsi 接口屏不需要在 sys\_config 中配置这组 pin，但同样会在此函数接口中打开与关闭对应的 pin。

Bon: 1: 为开，0: 为配置成 disable 状态。

**函数:** sunxi\_lcd\_dsi\_clk\_enable / sunxi\_lcd\_dsi\_clk\_disble

功能: dsi 接口屏使用, 使能/关闭 dsi 输出的 clk 信号。

原型: s32 sunxi\_lcd\_dsi\_clk\_enable(u32 scree\_id);

s32 sunxi\_lcd\_dsi\_clk\_disable(u32 scree\_id);

### 6.3. 屏的初始化

一部分屏需要进行初始化操作, 在开屏步骤函数中, 对应于 LCD\_panel\_init 函数, 提供了几种方式对屏的初始化。

对于 DSI 屏, 是通过 DSI-D0 通道进行初始化。对于 CPU 屏, 是通过 8080 总线的方式, 使用的是 LCDIO (PD,PH) 进行初始化。这种初始化方式, 其总线的引脚位置定义与 CPU 屏一致。

#### 6.3.1. MIPI DSI 屏的初始化

MIPI DSI 屏, 大部分需要初始化, 使用的是 DSI-D0 通道的 LP 模式进行初始化。提供的接口函数说明如下:

**函数:** sunxi\_lcd\_dsi\_dcs\_wr

功能: 对屏的 dcs 写操作

原型: \_\_s32 sunxi\_lcd\_dsi\_dcs\_wr(\_\_u32 sel, \_\_u8 cmd, \_\_u8\* para\_p, \_\_u32 para\_num);

参数说明:

cmd: dcs 写命令内容

para\_p: dcs 写命令的参数起始地址

para\_num: dcs 写命令的参数个数, 单位为 byte

**函数:** sunxi\_lcd\_dsi\_dcs\_wr\_2para

功能: 对屏的 dcs 写操作, 该命令带有两个参数

原型: \_\_s32 sunxi\_lcd\_dsi\_dcs\_wr\_2para(\_\_u32 sel, \_\_u8 cmd, \_\_u8 para1, \_\_u8 para2);

参数说明:

cmd: dcs 写命令内容

para1: dcs 写命令的第一个参数内容

para2: dcs 写命令的第二个参数内容

dsi\_dcs\_wr\_0para , dsi\_dcs\_wr\_1para , dsi\_dcs\_wr\_3para , dsi\_dcs\_wr\_4para , dsi\_dcs\_wr\_5para 定义与 dsi\_dcs\_wr\_2para 类似。

#### 6.3.2. CPU/I80 屏的初始化

CPU 屏的初始化可以参考“附录 5.3.5”的实例。

显示驱动提供 5 个接口函数可供使用。如下:

**函数:** sunxi\_lcd\_cpu\_write

功能：设定 CPU 屏的指定寄存器为指定的值

原型：void sunxi\_lcd\_cpu\_write(\_\_u32 sel, \_\_u32 index, \_\_u32 data)

函数内容为

```
Void sunxi_lcd_cpu_write(__u32 sel, __u32 index, __u32 data)
{
    Sunxi_lcd_cpu_write_index(sel, index);
    Sunxi_lcd_cpu_wirte_data(sel, data);
}
```

实现了 8080 总线上的两个写操作。

Sunxi\_lcd\_cpu\_write\_index 实现第一个写操作，这时 PIN 脚 RS (A1) 为低电平，总线数据上的数据内容为参数 index 的值。

Sunxi\_lcd\_cpu\_wirte\_data 实现第二个写操作，这时 PIN 脚 RS (A1) 为高电平，总线数据上的数据内容为参数 data 的值。

#### **函数：Sunxi\_lcd\_cpu\_write\_index**

功能：设定 CPU 屏为指定寄存器

原型：void Sunxi\_lcd\_cpu\_write\_index(\_\_u32 sel, \_\_u32 index);

具体说明见 Sunxi\_lcd\_cpu\_write。

#### **函数：Sunxi\_lcd\_cpu\_write\_data**

功能：设定 CPU 屏寄存器的值为指定的值

原型：void Sunxi\_lcd\_cpu\_write\_data(\_\_u32 sel, \_\_u32 data);

具体说明见 Sunxi\_lcd\_cpu\_write。

### 6.3.3. 使用 IO 模拟串行接口初始化

IO 的位置 (PIN 脚) 定义，默认属性 (输入输出) 定义及默认输出值在 sys\_config.fex。

见 5.4.4 lcd\_gpio\_x

显示驱动提供 2 个接口函数可供使用。说明如下：

函数：sunxi\_lcd\_gpio\_set\_value

功能：LCD\_GPIO PIN 脚上输出高电平或低电平

原型：s32 sunxi\_lcd\_gpio\_set\_value(u32 screen\_id, u32 io\_index, u32 value);

参数说明：

io\_index = 0: 对应于 sys\_config.fex 中的 lcd\_gpio\_0

io\_index = 1: 对应于 sys\_config.fex 中的 lcd\_gpio\_1

io\_index = 2: 对应于 sys\_config.fex 中的 lcd\_gpio\_2

io\_index = 3: 对应于 sys\_config.fex 中的 lcd\_gpio\_3

value = 0: 对应 IO 输出低电平

Value = 1: 对应 IO 输出高电平

只用于该 GPIO 定义为输出的情形。

函数：sunxi\_lcd\_gpio\_set\_direction

功能：设置 LCD\_GPIO PIN 脚为输入或输出模式

原型: `s32 sunxi_lcd_gpio_set_direction(u32 screen_id, u32 io_index, u32 direction);`

参数说明:

`io_index = 0`: 对应于 `sys_config.fex` 中的 `lcd_gpio_0`

`io_index = 1`: 对应于 `sys_config.fex` 中的 `lcd_gpio_1`

`io_index = 2`: 对应于 `sys_config.fex` 中的 `lcd_gpio_2`

`io_index = 3`: 对应于 `sys_config.fex` 中的 `lcd_gpio_3`

`direction = 0`: 对应 IO 设置为输入

`direction = 1`: 对应 IO 设置为输出

#### 6.3.4. 使用 `iic/spi` 串行接口初始化

需要在屏驱动中注册 `iic/spi` 设备对串行接口的访问。

Confidential