文档编号：

版本：V1.0

生效日期：2018-01-17

深圳市通力科技开发有限公司

TCL通力电子（惠州)有限公司

西安TCL软件开发有限公司



HI3516CV500 导入 mipi LCD设备方案详细介绍

|  |  |  |  |
| --- | --- | --- | --- |
| **文档作者：** | **丘伟盛** | **日期：** | 2020-5-10 |
| **项目经理：** |  | **日期：** |  |
| **审核：** |  | **日期：** |  |
| **批准：** |  | **日期：** |  |

**通力电子股份有限公司 版权所有**

**内部资料 注意保密**

文档修订记录

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **序号** | **版本号** | **变化状态** | **变更（+/-）说明** | **作者** | **日期** |
| 1 | V0.1 | C | 初版 | 丘伟盛 | 2020-5-10 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

\*变化状态：C――创建，A——增加，M——修改，D——删除

目录

[1. 简介 4](#_Toc6001)

[1.1. 文档目的 4](#_Toc15602)

[1.2. 适用范围 4](#_Toc8141)

[1.3. 术语和缩略语 4](#_Toc8594)

[1.4. 参考资料 4](#_Toc28662)

[2. 总体方案设计 5](#_Toc20902)

[2.1. 产品架构图 5](#_Toc11825)

[2.2. 软件架构图 6](#_Toc6959)

[2.3. MIPI屏基本框架： 7](#_Toc30217)

[2.4. Mipi屏软件兼容设计： 7](#_Toc17135)

[2.5. 平台系统层次图： 9](#_Toc15119)

[2.6. 项目LCD图像显示时序图： 9](#_Toc4259)

[3. Uboot阶段显示图像导入设计 10](#_Toc1723)

[3.1. 功能 10](#_Toc28140)

[3.2. 功能流程图 11](#_Toc4641)

[3.2.1. Uboot阶段LCD启动时序图： 11](#_Toc13720)

[3.3. 操作顺序说明（操作起始目录皆为uboot源码起始目录） 12](#_Toc671)

[3.3.1. 在UBOOT中加入解码，VO显示等相关命令 12](#_Toc32307)

[3.3.2. 设置UBOOT状态下相关数据结构 14](#_Toc2224)

[3.3.3. 设置UBOOT状态下相关接口函数： 22](#_Toc16891)

[3.3.4. 命令行执行示例： 25](#_Toc23167)

[4. 应用阶段显示图像导入设计 26](#_Toc6072)

[4.1. 功能 26](#_Toc4083)

[4.2. 功能流程图 27](#_Toc2816)

[4.2.1. 应用阶段LCD启动时序图： 27](#_Toc21582)

[4.2.2. 海思平台媒体处理流程图： 28](#_Toc22513)

[4.2.3. MIPI屏配置流程图： 28](#_Toc16791)

[4.3. 操作顺序说明（操作起始目录皆为SDK BSP代码起始目录） 29](#_Toc11623)

[4.3.1. 设置VO模块相关数据结构 29](#_Toc7262)

[4.3.2. 设置VO模块相关接口函数： 35](#_Toc29239)

[5. 系统出错处理 41](#_Toc6815)

# 简介

## 文档目的

本文档介绍了钉钉M2N方案采用Hi3516cv500主芯片进行在Uboot阶段以及应用阶段实现图片显示的流程， 包含函数接口和数据结构。为后续的详细开发设计提供参考和依据；

本文档主要适用于嵌入式Linux软件开发工程师, 用于流媒体/音视频编解码相关项目及领域.

## 适用范围

*[钉钉M2N开发，维护，设计软件方面人员*

## 术语和缩略语

| **缩略语/术语** | **全称** | **说明** |
| --- | --- | --- |
| MIPI | Mobile Industry Processor Interface | 移动行业处理器接口 |
| VO | Video OUT | 媒体输出 |
| Uboot |  | 嵌入式系统的引导加载程序 |

## 参考资料

《Hi3516CV500 专业型 Smart IP Camera SoC用户指南.pdf》

《HiMPP V4.0 媒体处理软件开发参考.pdf》

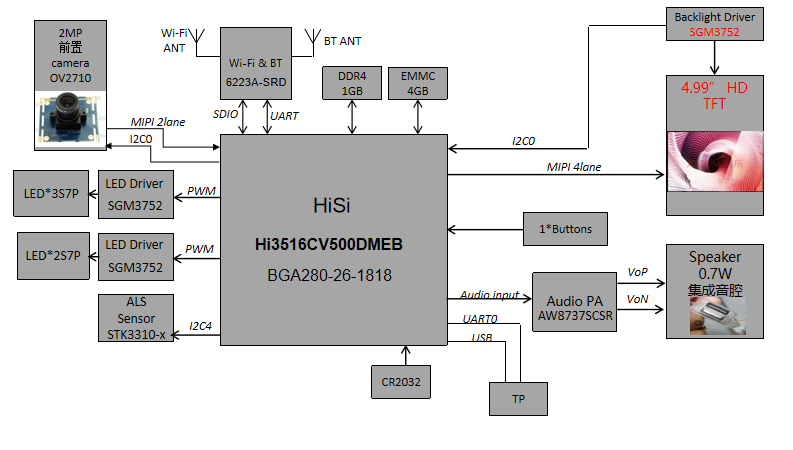
《Hi3516CV500╱Hi3516DV300╱Hi3516AV300 U-boot 移植应用开发指南.pdf》

《开机画面使用指南.pdf》

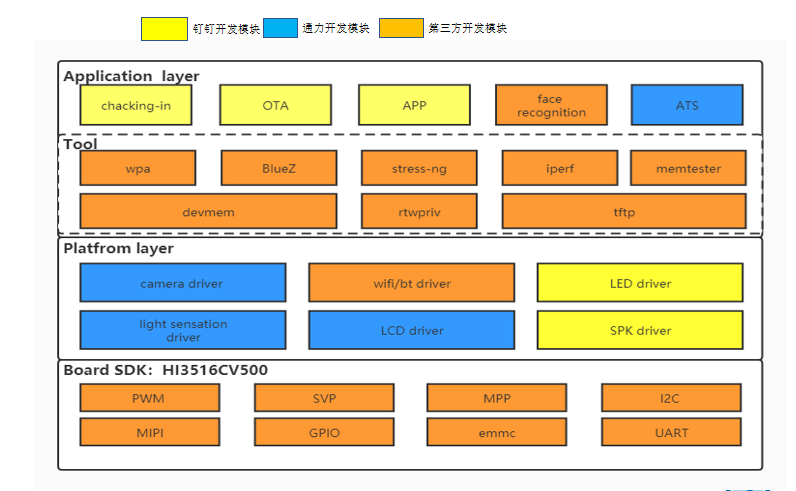
《MIPI使用指南.pdf》

# 总体方案设计

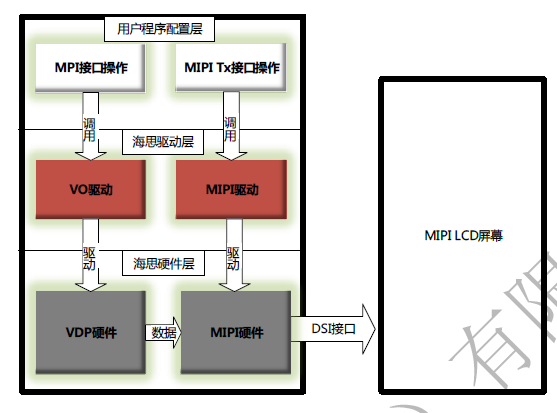
## 产品架构图



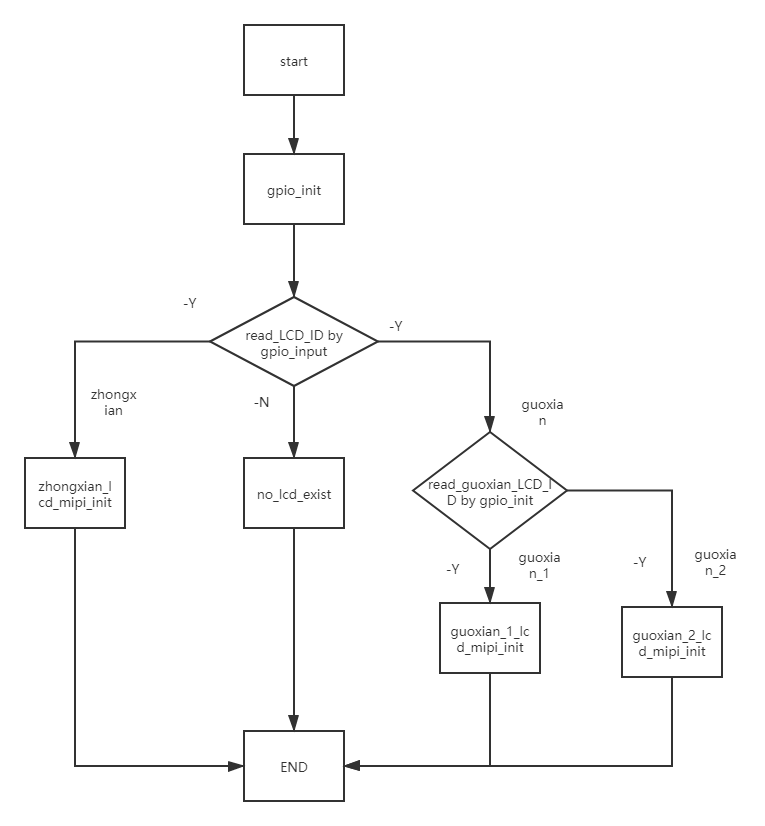
## 软件架构图



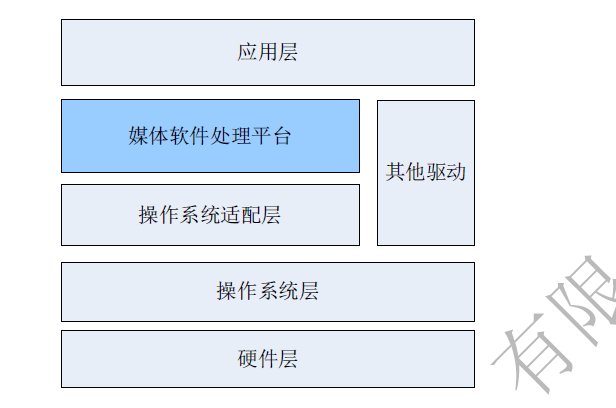
## MIPI屏基本框架：



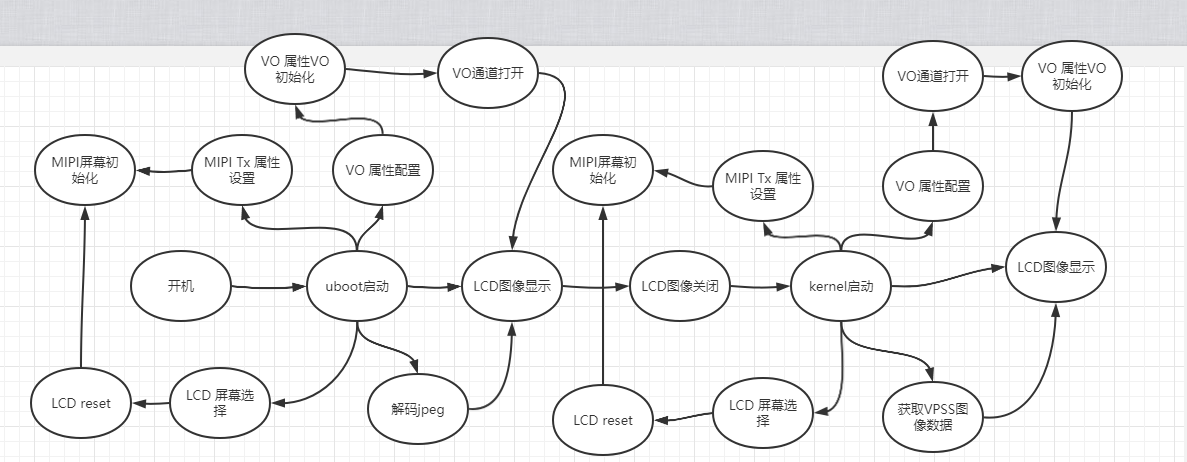
## Mipi屏软件兼容设计：



## 平台系统层次图：



## 项目LCD图像显示时序图：



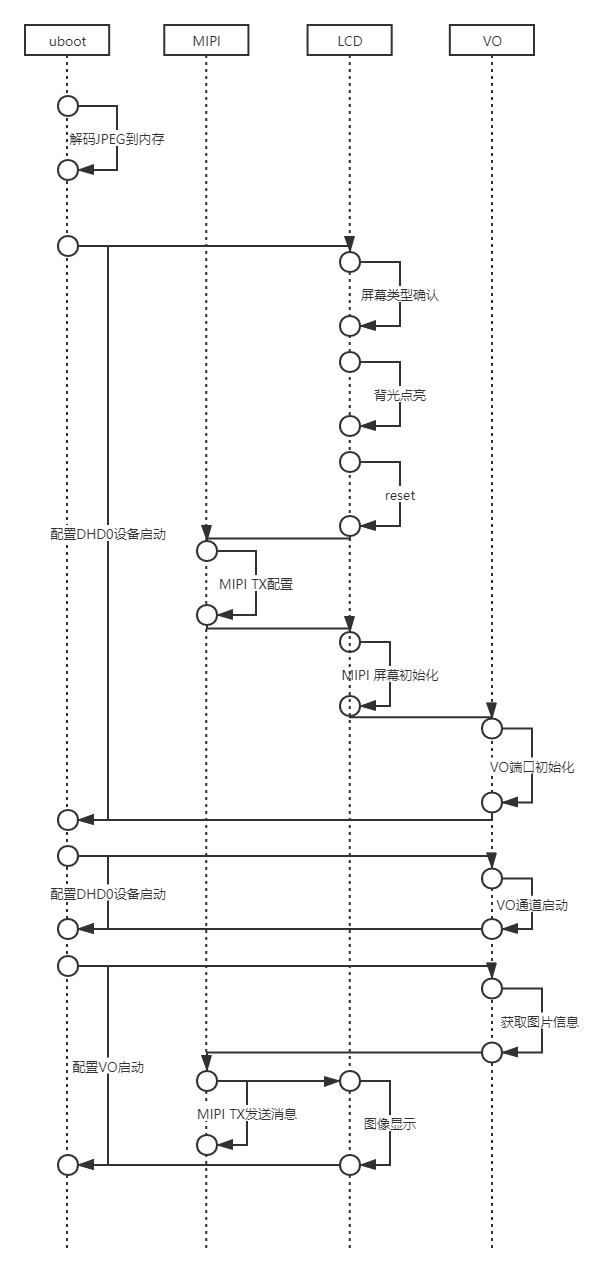
# Uboot阶段显示图像导入设计

## 功能

*实现jepg图像在对应类型mipi屏幕中显示*

## 功能流程图

### Uboot阶段LCD启动时序图：

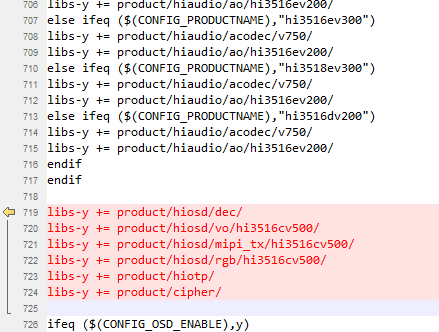


该时序图讲述UBOOT在启动LCD显示图像中，mipi TX，VO模块，LCD模组之间的时序交互逻辑。

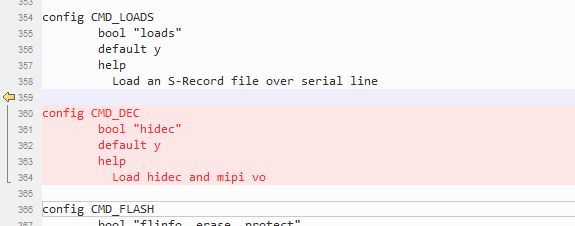
## 操作顺序说明（操作起始目录皆为uboot源码起始目录）

### 在UBOOT中加入解码，VO显示等相关命令

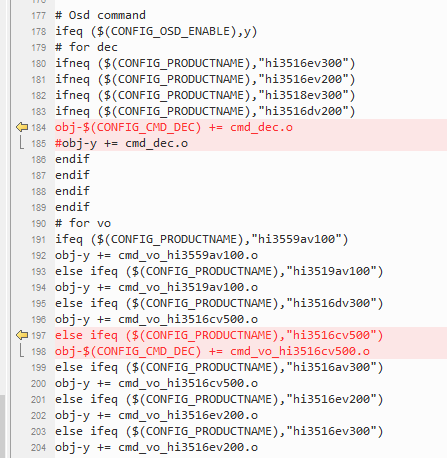
#### 在uboot源码目录下的makefile加入以下内容指定平台为3516cv500.



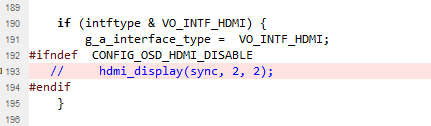
#### 在cmd/Kconfig 文件下追加CMD\_DEC命令

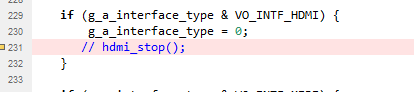


#### 在cmd/makefile文件下将cmd\_dec.c链接进来：

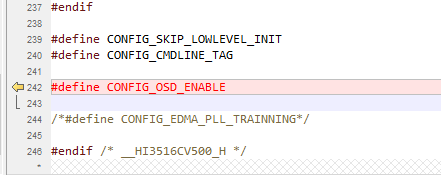


#### 在cmd/cmd\_vo\_hi3516cv500.c文件下将hdmi相关函数屏蔽（项目没有hdmi需求）：

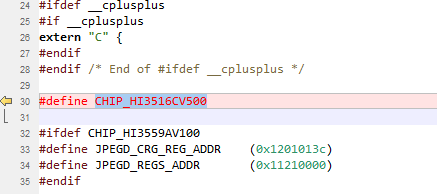




#### 在include/configs/hi3516cv500.h文件中把CONFIG\_OSD\_ENABLE 选项打开：



#### 在product/hiosd/dec/jpegd\_drv.h文件中把#define CHIP\_HI3516CV500 定义出来，用以指定平台为3516CV500：



#### 以上工序完成之后重新编译uboot平且烧录到HI3516CV500中，即可使用VO等相关功能。

### 设置UBOOT状态下相关数据结构

#### 在product\hiosd\vo\hi3516cv500\vou\_drv.c下配置相关内容：

##### 枚举类型1：

typedef enum {

LCD\_IS\_NOT\_PREESNT = 0x0,

LCD\_VENDOR\_GUOXIAN\_1 = 0x1,

LCD\_VENDOR\_GUOXIAN\_2 = 0x2,

LCD\_VENDOR\_ZHONGXIAN\_1 = 0x3,

LCD\_VENDOR\_ZHONGXIAN\_2 = 0x4,

} LCD\_VENDOR\_E;

解析：屏幕类型枚举

##### 结构体类型1、：

vo\_user\_intfsync\_info g\_user\_intfsync\_info =

{

.clk\_reverse = HI\_FALSE,

.user\_intfsync\_attr = {

.clk\_source = VO\_CLK\_SOURCE\_PLL,

.user\_sync\_pll = {

.fbdiv = 69,

.frac = 0xd8e219,

.refdiv = 2,

.postdiv1 = 4,

.postdiv2 = 3

},

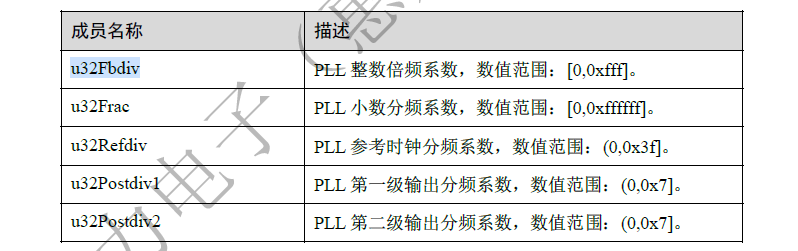
},

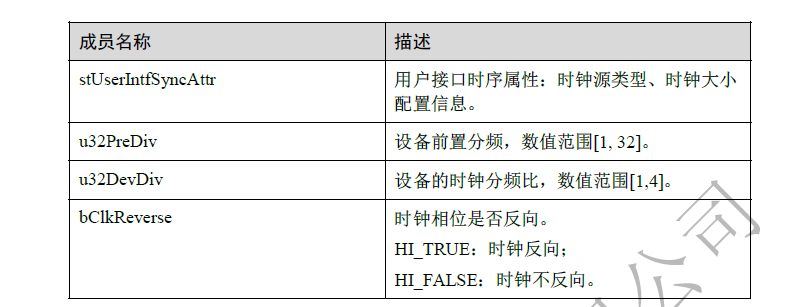
.pre\_div = 1,

.dev\_div = 1,

};

解析：





##### 结构体类型2、：

static hal\_dev\_config g\_hal\_dev\_cfg[VO\_DEV\_MAX\_NUM] = {

{

.enable = HI\_FALSE,

.bk\_grd = VO\_BACKGROUD\_DEFAULT,

.intf\_type = VO\_INTF\_MIPI,

.out\_sync = VO\_OUTPUT\_USER, /\* for debug: VO\_OUTPUT\_800x600\_60 \*/

.pixel\_fmt = HAL\_INPUTFMT\_RGB\_888, /\* VOU\_PIXERL\_FORMAT\_RGB888, VOU\_PIXERL\_FORMAT\_YCBCR422 \*/

//.pixel\_fmt = HAL\_INPUTFMT\_Y\_CB\_CR\_SEMIPLANAR\_422

}

};

解析：

Enable ： 使能开关

bk\_grd ： 背景颜色

intf\_type ： VO接口类型

out\_sync ： VO接口时序类型

pixel\_fmt ： 图像格式类型

##### 数组类型1、

hal\_disp\_syncinfo g\_sync\_timing[VO\_OUTPUT\_BUTT] = {

/\* synm, iop, itf, vact, vbb, vfb, hact, hbb, hfb, hmid,bvact,bvbb, bvfb, hpw, vpw,idv, ihs, ivs \*/

{ 0, 0, 0, 288, 22, 2, 720, 132, 12, 1, 288, 23, 2, 126, 3, 0, 0, 0 }, /\* 576I(PAL) \*/

{ 0, 0, 0, 240, 18, 4, 720, 119, 19, 1, 240, 19, 4, 124, 3, 0, 0, 0 }, /\* 480I(NTSC) \*/

{ 0, 1, 1, 1080, 41, 4, 1920, 192, 638, 1, 1, 1, 1, 44, 5, 0, 0, 0 }, /\* 1080P@24Hz \*/

{ 0, 1, 1, 1080, 41, 4, 1920, 192, 528, 1, 1, 1, 1, 44, 5, 0, 0, 0 }, /\* 1080P@25Hz \*/

{ 0, 1, 1, 1080, 41, 4, 1920, 192, 88, 1, 1, 1, 1, 44, 5, 0, 0, 0 }, /\* 1080P@30Hz \*/

{ 0, 1, 1, 720, 25, 5, 1280, 260, 440, 1, 1, 1, 1, 40, 5, 0, 0, 0 }, /\* 720P@50Hz \*/

{ 0, 1, 1, 720, 25, 5, 1280, 260, 110, 1, 1, 1, 1, 40, 5, 0, 0, 0 }, /\* 720P@60Hz \*/

{ 0, 0, 1, 540, 20, 2, 1920, 192, 528, 1128, 540, 21, 2, 44, 5, 0, 0, 0 }, /\* 1080I@50Hz \*/

{ 0, 0, 1, 540, 20, 2, 1920, 192, 88, 908, 540, 21, 2, 44, 5, 0, 0, 0 }, /\* 1080I@60Hz \*/

{ 0, 1, 1, 1080, 41, 4, 1920, 192, 528, 1, 1, 1, 1, 44, 5, 0, 0, 0 }, /\* 1080P@50Hz \*/

{ 0, 1, 1, 1080, 41, 4, 1920, 192, 88, 1, 1, 1, 1, 44, 5, 0, 0, 0 }, /\* 1080P@60Hz \*/

{ 1, 1, 1, 576, 44, 5, 720, 132, 12, 1, 1, 1, 1, 64, 5, 0, 0, 0 }, /\* 576P@50Hz \*/

{ 1, 1, 1, 480, 36, 9, 720, 122, 16, 1, 1, 1, 1, 62, 6, 0, 0, 0 }, /\* 480P@60Hz \*/

{ 1, 1, 2, 600, 27, 1, 800, 216, 40, 1, 1, 1, 1, 128, 4, 0, 0, 0 }, /\* 800\*600@60Hz VGA@60Hz \*/

{ 1, 1, 2, 768, 35, 3, 1024, 296, 24, 1, 1, 1, 1, 136, 6, 0, 1, 1 }, /\* 1024x768@60Hz \*/

{ 1, 1, 2, 1024, 41, 1, 1280, 360, 48, 1, 1, 1, 1, 112, 3, 0, 0, 0 }, /\* 1280x1024@60Hz \*/

{ 1, 1, 2, 768, 27, 3, 1366, 356, 70, 1, 1, 1, 1, 143, 3, 0, 0, 0 }, /\* 1366x768@60Hz \*/

{ 1, 1, 2, 900, 31, 3, 1440, 384, 80, 1, 1, 1, 1, 152, 6, 0, 1, 0 }, /\* 1440x900@60Hz \*/

{ 1, 1, 2, 800, 28, 3, 1280, 328, 72, 1, 1, 1, 1, 128, 6, 0, 1, 0 }, /\* 1280\*800@60Hz VGA@60Hz \*/

{ 1, 1, 2, 1200, 49, 1, 1600, 496, 64, 1, 1, 1, 1, 192, 3, 0, 0, 0 }, /\* 1600\*1200@60Hz \*/

{ 1, 1, 2, 1050, 36, 3, 1680, 456, 104, 1, 1, 1, 1, 176, 6, 0, 1, 0 }, /\* 1680\*1050@60Hz \*/

/\* 1920\*1200@60Hz CVT (reduced blanking) \*/

{ 1, 1, 2, 1200, 32, 3, 1920, 112, 48, 1, 1, 1, 1, 32, 6, 0, 0, 1 },

{ 1, 1, 2, 480, 35, 10, 640, 144, 16, 1, 1, 1, 1, 96, 2, 0, 1, 1 }, /\* 640\*480@60Hz CVT \*/

{ 0, 0, 0, 288, 22, 2, 960, 176, 16, 1, 288, 23, 2, 168, 3, 0, 0, 0 }, /\* 960H(PAL) \*/

{ 0, 0, 0, 240, 18, 4, 960, 163, 21, 1, 240, 19, 4, 168, 3, 0, 0, 0 }, /\* 960H(NTSC) \*/

{ 0, 1, 1, 2160, 72, 8, 1920, 192, 88, 1, 1, 1, 1, 44, 5, 0, 0, 0 }, /\* 1920\*2160@30Hz \*/

{ 1, 1, 2, 1440, 39, 2, 2560, 112, 48, 1, 1, 1, 1, 32, 5, 0, 0, 0 }, /\* 2560\*1440@30Hz \*/

{ 1, 1, 2, 1440, 39, 2, 2560, 112, 48, 1, 1, 1, 1, 32, 5, 0, 0, 0 }, /\* 2560\*1440@60Hz \*/

/\* 2560\*1600@60Hz CVT (reduced blanking) \*/

{ 0, 1, 2, 1600, 43, 3, 2560, 112, 48, 1, 1, 1, 1, 32, 6, 0, 0, 1 },

{ 0, 1, 1, 2160, 82, 8, 3840, 384, 1276, 1, 1, 1, 1, 88, 10, 0, 0, 0 }, /\* 3840\*2160@24Hz \*/

{ 0, 1, 1, 2160, 82, 8, 3840, 384, 1056, 1, 1, 1, 1, 88, 10, 0, 0, 0 }, /\* 3840\*2160@25Hz \*/

{ 0, 1, 1, 2160, 82, 8, 3840, 384, 176, 1, 1, 1, 1, 88, 10, 0, 0, 0 }, /\* 3840\*2160@30Hz \*/

{ 0, 1, 1, 2160, 82, 8, 3840, 384, 1056, 1, 1, 1, 1, 88, 10, 0, 0, 0 }, /\* 3840\*2160@50Hz \*/

{ 0, 1, 1, 2160, 82, 8, 3840, 384, 176, 1, 1, 1, 1, 88, 10, 0, 0, 0 }, /\* 3840\*2160@60Hz \*/

{ 0, 1, 1, 2160, 82, 8, 4096, 384, 1020, 1, 1, 1, 1, 88, 10, 0, 0, 0 }, /\* 4096x2160@24 \*/

{ 0, 1, 1, 2160, 82, 8, 4096, 216, 968, 1, 1, 1, 1, 88, 10, 0, 0, 0 }, /\* 4096x2160@25 \*/

{ 0, 1, 1, 2160, 82, 8, 4096, 216, 88, 1, 1, 1, 1, 88, 10, 0, 0, 0 }, /\* 4096x2160@30 \*/

{ 0, 1, 1, 2160, 82, 8, 4096, 216, 968, 1, 1, 1, 1, 88, 10, 0, 0, 0 }, /\* 4096x2160@50 \*/

{ 0, 1, 1, 2160, 82, 8, 4096, 216, 88, 1, 1, 1, 1, 88, 10, 0, 0, 0 }, /\* 4096x2160@60 \*/

{ 0, 1, 1, 240, 15, 9, 320, 65, 7, 1, 240, 14, 9, 1, 1, 0, 0, 0 }, /\* 320X240@60 8bit LCD \*/

{ 0, 1, 1, 240, 2, 2, 320, 5, 10, 1, 1, 1, 1, 10, 1, 0, 0, 0 }, /\* 320X240@50 6bit LCD \*/

{ 0, 1, 1, 320, 10, 4, 240, 30, 10, 1, 1, 1, 1, 10, 2, 0, 0, 0 }, /\* 240X320@50 6bit LCD \*/

{ 0, 1, 1, 320, 2, 2, 240, 20, 10, 1, 1, 1, 1, 2, 1, 0, 0, 0 }, /\* 240X320@60 16bit LCD \*/

{ 0, 1, 1, 600, 23, 12, 800, 210, 46, 1, 1, 1, 1, 2, 1, 0, 0, 0 }, /\* 800X600@60 24bit LCD \*/

{ 1, 1, 1, 1280, 20, 8, 720, 80, 90, 1, 1, 1, 1, 20, 6, 0, 0, 0 }, /\* 720 x1280 at 60 hz \*/

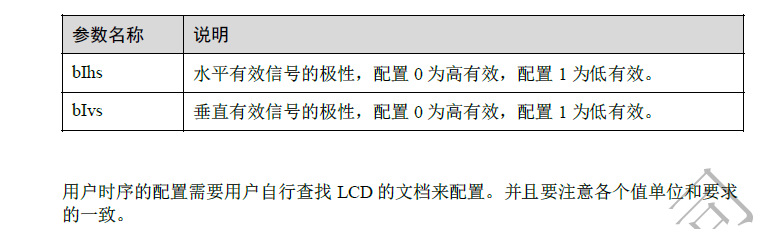
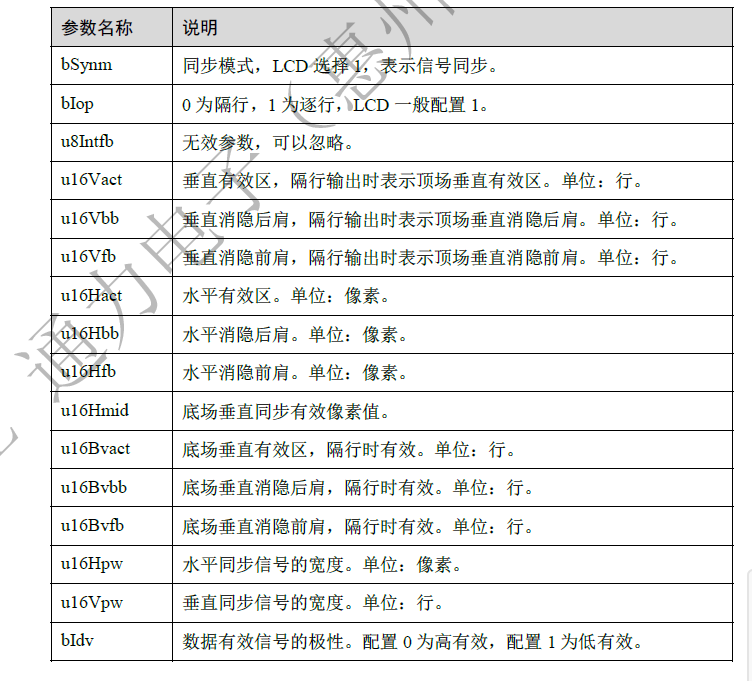
{ 0, 1, 1, 1920, 36, 16, 1080, 28, 130, 1, 1, 1, 1, 24, 4, 0, 0, 0 }, /\* 1080 x1920 at 60 hz \*/

{ 0, 1, 1, 4320, 64, 16, 7680, 768, 552, 1, 1, 1, 1, 176, 20, 0, 0, 0 }, /\* 7680x4320@30 \*/

{ 1, 1, 0, 1280, 18, 16, 720, 60, 40, 1, 1, 1, 1, 20, 4, 0, 0, 0 }, /\* user \*/

};

解析：



#### 在product\hiosd\mipi\_tx\hi3516cv500\mip\_tx.c下配置相关内容：

##### 结构体类型1、

combo\_dev\_cfg\_t g\_mipi\_tx\_user\_config\_zhongxian = {

.devno = 0,

.lane\_id = {0, 1, 2, 3},

.output\_mode = OUTPUT\_MODE\_DSI\_VIDEO,

.output\_format = OUT\_FORMAT\_RGB\_24\_BIT,

.video\_mode = BURST\_MODE,

.sync\_info = {

.vid\_pkt\_size = 720, // hact

.vid\_hsa\_pixels = 20, // hsa

.vid\_hbp\_pixels = 60, // hbp

.vid\_hline\_pixels = 890, // hact + hsa + hbp + hfp

.vid\_vsa\_lines = 6, // vsa

.vid\_vbp\_lines = 14, // vbp

.vid\_vfp\_lines = 8, // vfp

.vid\_active\_lines = 1280,// vact

.edpi\_cmd\_size = 0,

},

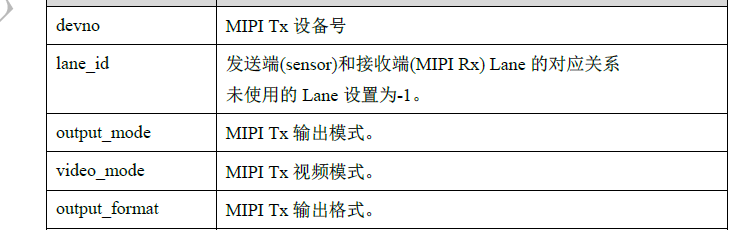
.phy\_data\_rate = 470,

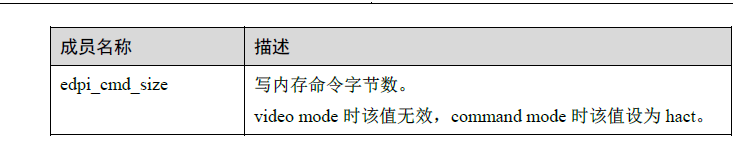
.pixel\_clk = 69848,

};

该结构体为屏幕的属性，原始参数可以找屏厂要，通过《RGB\_MIPI屏幕时钟时序计算器.xlsx》可以获得属性值，按需增加结构体，输入对应参数即可。

解析：





### 设置UBOOT状态下相关接口函数：

#### 在product\hiosd\mipi\_tx\hi3516cv500\mip\_tx.c下配置相关内容：

##### lcd\_detect函数

|  |  |
| --- | --- |
| 函数原型 | **static int lcd\_detective(void)** |
| 功能描述 | Mipi 屏幕类型检测 |
| 输入参数 | 无 |
| 输出参数 | MIPI屏幕类型 |
| 全局变量值的变化 | g\_current\_lcd\_vendor = lcd\_detective(); |
| 返回值 | Int型返回值 |

函数实现描述（自然语言）：

static int lcd\_detective(void)

{

屏幕类型检测

}

##### LCD\_Backlight\_on函数

|  |  |
| --- | --- |
| 函数原型 | **void LCD\_Backlight\_on(int on)** |
| 功能描述 | 屏幕背光控制 |
| 输入参数 | Int型 0 或 1 |
| 输出参数 | 无 |
| 全局变量值的变化 | 无 |
| 返回值 | 无 |

函数实现描述（自然语言）：

void LCD\_Backlight\_on(int on)

{

If（0）

{

关闭背光；

}

Else if（1）

{

打开背光；

}

}

##### SAMPLE\_PRIVATE\_VO\_InitScreen720x1280\_guoxian\_1函数(该类函数依屏幕初始化要求调整)

|  |  |
| --- | --- |
| 函数原型 | **static int SAMPLE\_PRIVATE\_VO\_InitScreen720x1280\_guoxian\_1(void)** |
| 功能描述 | 国显1mipi初始化 |
| 输入参数 | 无 |
| 输出参数 | Int型返回值 |
| 全局变量值的变化 | 无 |
| 返回值 | Int型返回值 |

函数实现描述（自然语言）：

static int SAMPLE\_PRIVATE\_VO\_InitScreen720x1280\_guoxian\_1(void)

{

初始化屏幕；

}

##### SAMPLE\_PRIVATE\_VO\_InitScreen720x1280\_guoxian\_2函数(该类函数依屏幕初始化要求调整)

|  |  |
| --- | --- |
| 函数原型 | **static int SAMPLE\_PRIVATE\_VO\_InitScreen720x1280\_guoxian\_2(void)** |
| 功能描述 | 国显2 mipi初始化 |
| 输入参数 | 无 |
| 输出参数 | Int型返回值 |
| 全局变量值的变化 | 无 |
| 返回值 | Int型返回值 |

函数实现描述（自然语言）：

static int SAMPLE\_PRIVATE\_VO\_InitScreen720x1280\_guoxian\_2(void)

{

初始化屏幕；

}

##### SAMPLE\_PRIVATE\_VO\_InitScreen720x1280\_zhongxian函数(该类函数依屏幕初始化要求调整)

|  |  |
| --- | --- |
| 函数原型 | **static int SAMPLE\_PRIVATE\_VO\_InitScreen720x1280\_**zhongxian**(void)** |
| 功能描述 | 中显1 mipi初始化 |
| 输入参数 | 无 |
| 输出参数 | Int型返回值 |
| 全局变量值的变化 | 无 |
| 返回值 | Int型返回值 |

函数实现描述（自然语言）：

static int SAMPLE\_PRIVATE\_VO\_InitScreen720x1280\_zhongxian(void)

{

初始化屏幕；

}

##### SAMPLE\_PRIVATE\_VO\_InitScreen720x1280\_zhongxian\_2函数(该类函数依屏幕初始化要求调整)

|  |  |
| --- | --- |
| 函数原型 | **static int SAMPLE\_PRIVATE\_VO\_InitScreen720x1280\_**zhongxian\_2**(void)** |
| 功能描述 | 中显2 mipi初始化 |
| 输入参数 | 无 |
| 输出参数 | Int型返回值 |
| 全局变量值的变化 | 无 |
| 返回值 | Int型返回值 |

函数实现描述（自然语言）：

static int SAMPLE\_PRIVATE\_VO\_InitScreen720x1280\_zhongxian\_2(void)

{

初始化屏幕；

}

##### LCD\_Reset函数

|  |  |
| --- | --- |
| 函数原型 | **void LCD\_Reset(void)** |
| 功能描述 | LCD 复位，清除数据 |
| 输入参数 | 无 |
| 输出参数 | 无 |
| 全局变量值的变化 | 无 |
| 返回值 | 无 |

函数实现描述（自然语言）：

void LCD\_Reset(void)

{

LCD复位；

}

##### SAMPLE\_USER\_INIT\_MIPITx函数

|  |  |
| --- | --- |
| 函数原型 | **static int SAMPLE\_USER\_INIT\_MIPITx(cmd\_info\_t \*pcmd\_info)** |
| 功能描述 | mipi数据传输 |
| 输入参数 | 数据地址 |
| 输出参数 | Int型返回值 |
| 全局变量值的变化 | 无 |
| 返回值 | Int型返回值 |

函数实现描述（自然语言）：

static int SAMPLE\_USER\_INIT\_MIPITx(cmd\_info\_t \*pcmd\_info)

{

数据传输；

}

### 命令行执行示例：

1. 设置环境变量，配置jpeg解码参数：

setenv jpeg\_addr 0x83000000

setenv jpeg\_size 0x4D500

setenv vobuf 0x81000000

setenv jpeg\_emar\_buf 0x85000000

1. 解码jpeg到内存：

decjpg 0 0

1. 配置DHD0设备启动：

startvo 0 16384 47

1. 配置VO启动：

startvl 0 0x81000000 768 0 0 720 1280

1. 关闭VO：

stopvl 0

1. 关闭DHD0设备：

stopvo 0

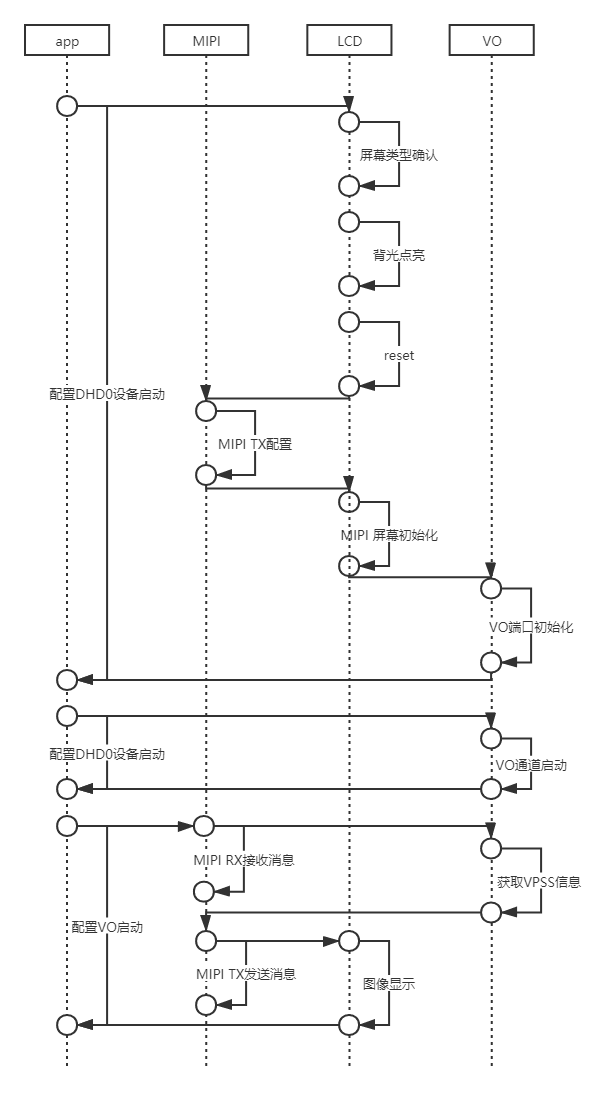
# 应用阶段显示图像导入设计

## 功能

*实现摄像头传输图像在对应类型mipi屏幕中显示*

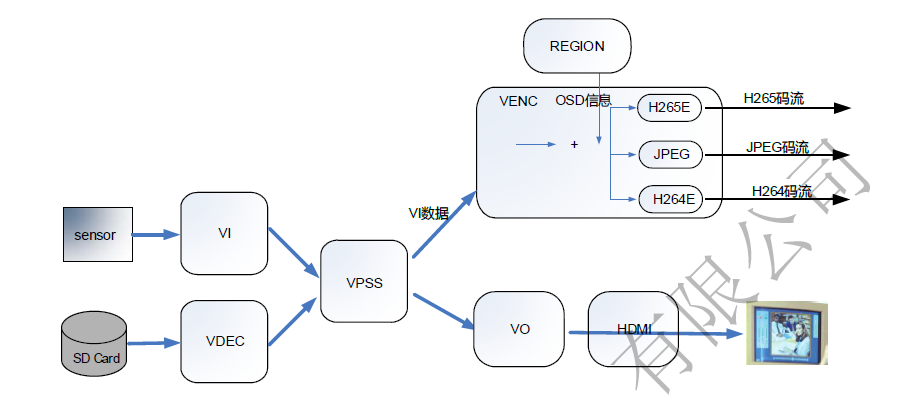
## 功能流程图

### 应用阶段LCD启动时序图：

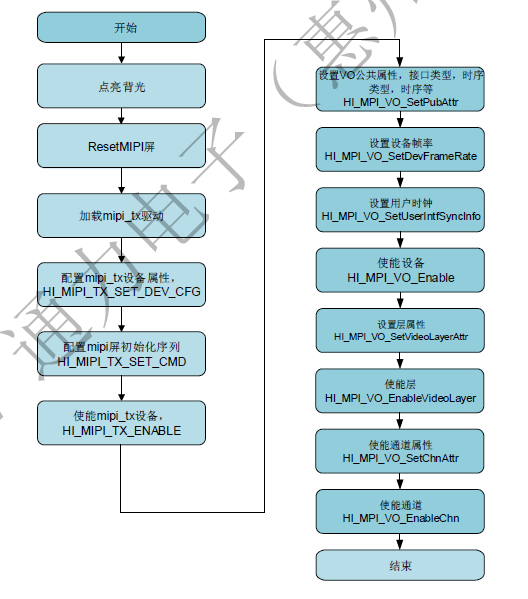


该时序图讲述UBOOT在启动LCD显示图像中，mipi ，VO模块，LCD模组之间的时序交互逻辑。

### 海思平台媒体处理流程图：



### MIPI屏配置流程图：



## 操作顺序说明（操作起始目录皆为SDK BSP代码起始目录）

### 设置VO模块相关数据结构

#### 在smp\a7\_linux\mpp\sample\common\sample\_commom\_vo.c下配置相关内容：

##### 枚举类型1、

typedef enum {

LCD\_IS\_NOT\_PREESNT = 0x0,

LCD\_VENDOR\_GUOXIAN\_1 = 0x1,

LCD\_VENDOR\_GUOXIAN\_2 = 0x2,

LCD\_VENDOR\_ZHONGXIAN\_1 = 0x3,

} LCD\_VENDOR\_E;

解析：屏幕类型枚举

##### 结构体类型1、

combo\_dev\_cfg\_t VO\_OUTPUT\_USER\_CONFIG\_ZHONGXIAN =

{

.devno = 0,

.lane\_id = {0, 1, 2, 3},

.output\_mode = OUTPUT\_MODE\_DSI\_VIDEO,

.output\_format = OUT\_FORMAT\_RGB\_24\_BIT,

.video\_mode = BURST\_MODE,

.sync\_info = {

.vid\_pkt\_size = 720, // hact

.vid\_hsa\_pixels = 20, // hsa

.vid\_hbp\_pixels = 60, // hbp

.vid\_hline\_pixels = 890, // hact + hsa + hbp + hfp

.vid\_vsa\_lines = 6, // vsa

.vid\_vbp\_lines = 14, // vbp

.vid\_vfp\_lines = 8, // vfp

.vid\_active\_lines = 1280,// vact

.edpi\_cmd\_size = 0,

},

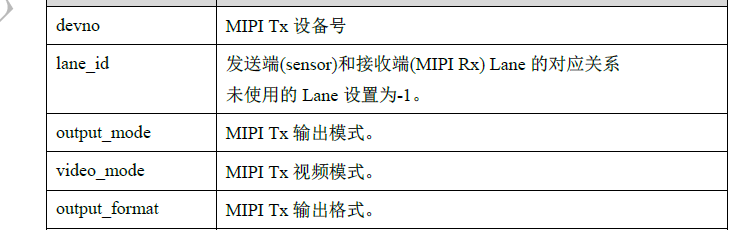
.phy\_data\_rate = 470,

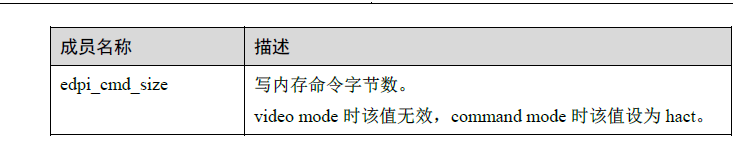
.pixel\_clk = 69848,

};

该结构体为屏幕的属性，原始参数可以找屏厂要，通过《RGB\_MIPI屏幕时钟时序计算器.xlsx》可以获得属性值，按需增加结构体，输入对应参数即可。

解析：





##### 结构体类型2、

typedef struct hiVO\_PUB\_ATTR\_S {

HI\_U32 u32BgColor; /\* RW; Background color of a device, in RGB format. \*/

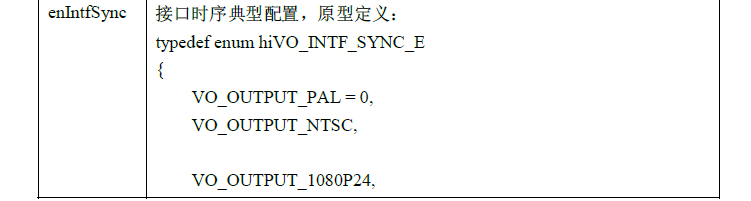
VO\_INTF\_TYPE\_E enIntfType; /\* RW; Type of a VO interface \*/

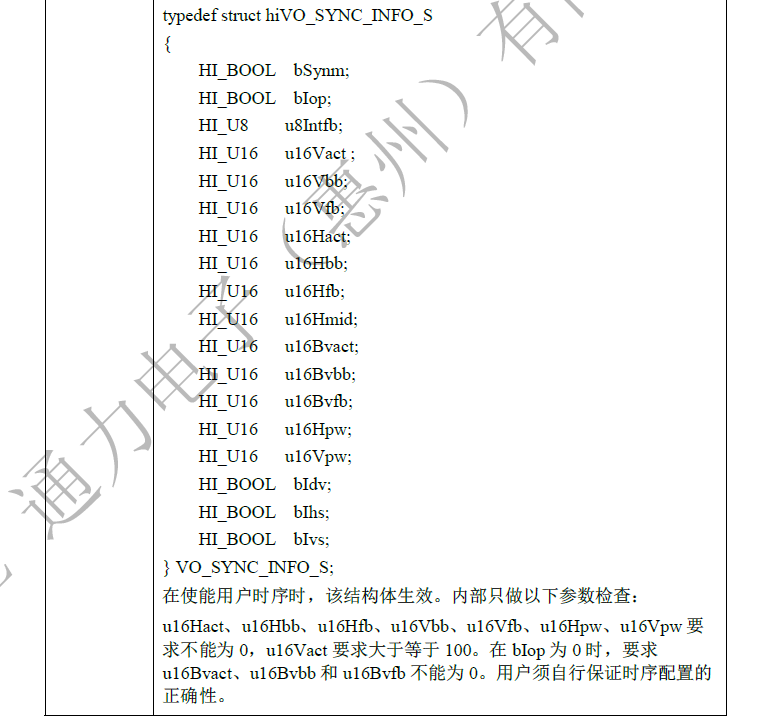
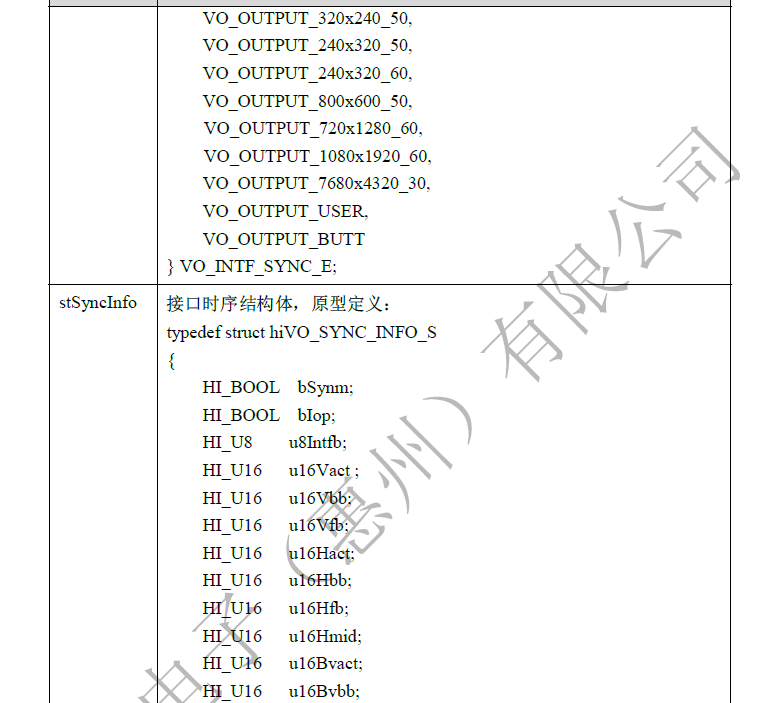
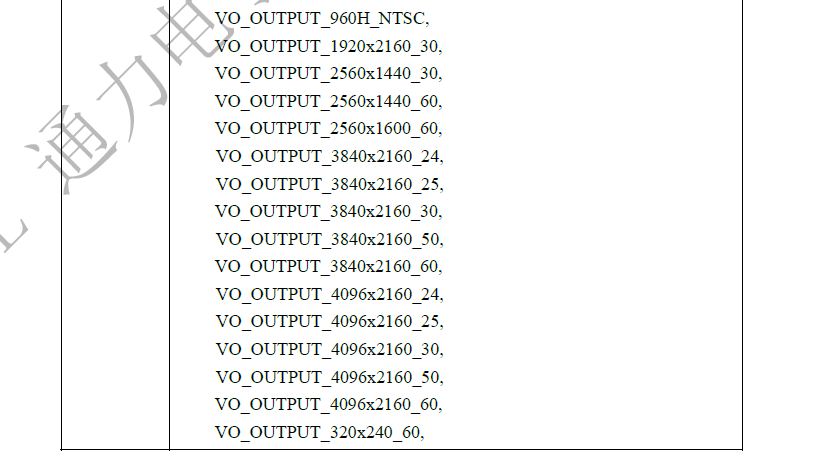
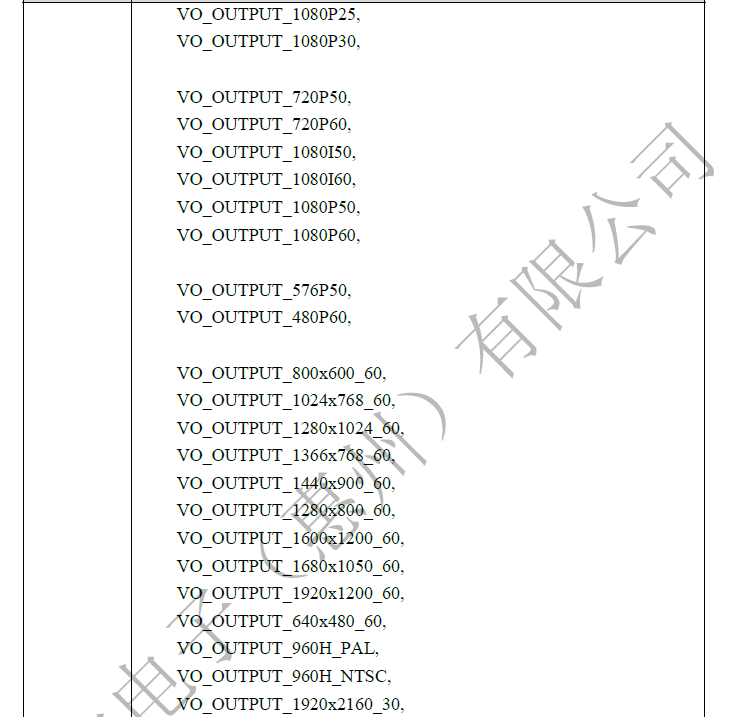
VO\_INTF\_SYNC\_E enIntfSync; /\* RW; Type of a VO interface timing \*/

VO\_SYNC\_INFO\_S stSyncInfo; /\* RW; Information about VO interface timings \*/

} VO\_PUB\_ATTR\_S;

解析：





将配置好的参数作为HI\_MPI\_VO\_SetPubAttr的输入参数。

##### 结构体类型3、

typedef struct hiVO\_USER\_INTFSYNC\_INFO\_S {

VO\_USER\_INTFSYNC\_ATTR\_S stUserIntfSyncAttr;

HI\_U32 u32PreDiv;

HI\_U32 u32DevDiv;

HI\_BOOL bClkReverse;

} VO\_USER\_INTFSYNC\_INFO\_S;

解析：



将配置好的参数作为HI\_MPI\_VO\_SetUserIntfSyncInfo的输入参数

### 设置VO模块相关接口函数：

#### 在smp\a7\_linux\mpp\sample\common\sample\_commom\_vo.c下配置相关内容：

##### lcd\_detect函数

|  |  |
| --- | --- |
| 函数原型 | **static int lcd\_detective(void)** |
| 功能描述 | Mipi 屏幕类型检测 |
| 输入参数 | 无 |
| 输出参数 | MIPI屏幕类型 |
| 全局变量值的变化 | g\_current\_lcd\_vendor = lcd\_detective(); |
| 返回值 | Int型返回值 |

函数实现描述（自然语言）：

static int lcd\_detective(void)

{

屏幕类型检测

}

##### gpio\_direction\_input函数

|  |  |
| --- | --- |
| 函数原型 | **static int gpio\_direction\_input(unsigned int gpio)** |
| 功能描述 | Mipi 屏幕ID引脚GPIO输入模式数值 |
| 输入参数 | GPIO |
| 输出参数 | Int型返回值 |
| 全局变量值的变化 | 无 |
| 返回值 | Int型返回值 |

函数实现描述（自然语言）：

static int gpio\_direction\_input(unsigned int gpio)

{

GPIO口配置成输入；

}

##### gpio\_direction\_output函数

|  |  |
| --- | --- |
| 函数原型 | **static int gpio\_direction\_output(unsigned int gpio, int value)** |
| 功能描述 | Mipi 屏幕ID引脚GPIO输出模式数值 |
| 输入参数 | GPIO |
| 输出参数 | Int型返回值 |
| 全局变量值的变化 | 无 |
| 返回值 | Int型返回值 |

函数实现描述（自然语言）：

static int gpio\_direction\_output(unsigned int gpio)

{

GPIO口配置成输出；

}

##### gpio\_get\_value函数

|  |  |
| --- | --- |
| 函数原型 | **static unsigned char gpio\_get\_value(unsigned int gpio)** |
| 功能描述 | Mipi 屏幕ID引脚GPIO数值 |
| 输入参数 | GPIO |
| 输出参数 | Int型返回值 |
| 全局变量值的变化 | 无 |
| 返回值 | Int型返回值 |

函数实现描述（自然语言）：

static int gpio\_get\_value(unsigned int gpio)

{

GPIO口数值输出；

}

##### SAMPLE\_COMM\_VO\_StartVO函数

|  |  |
| --- | --- |
| 函数原型 | **HI\_S32 SAMPLE\_COMM\_VO\_StartVO(SAMPLE\_VO\_CONFIG\_S \*pstVoConfig)** |
| 功能描述 | VO模块启动 |
| 输入参数 | VO配置项 |
| 输出参数 | HI\_S32型返回值 |
| 全局变量值的变化 | g\_current\_lcd\_vendor = lcd\_detective(); |
| 返回值 | HI\_S32型返回值 |

函数实现描述（自然语言）：

HI\_S32 SAMPLE\_COMM\_VO\_StartVO(SAMPLE\_VO\_CONFIG\_S \*pstVoConfig)

{

屏幕类型检测；

MIPI设备属性设置；

VO用户时序参数值配置；

V0用户时钟参数值配置；

调用HI\_MPI\_VO\_SetPubAttr和HI\_MPI\_VO\_SetUserIntfSyncInfo接口；

}

##### LCD\_Backlight\_on函数

|  |  |
| --- | --- |
| 函数原型 | **void LCD\_Backlight\_on(int on)** |
| 功能描述 | 屏幕背光控制 |
| 输入参数 | Int型 0 或 1 |
| 输出参数 | 无 |
| 全局变量值的变化 | 无 |
| 返回值 | 无 |

函数实现描述（自然语言）：

void LCD\_Backlight\_on(int on)

{

If（0）

{

关闭背光；

}

Else if（1）

{

打开背光；

}

}

##### SAMPLE\_PRIVATE\_VO\_InitScreen720x1280\_guoxian\_1函数(该类函数依屏幕初始化要求调整)

|  |  |
| --- | --- |
| 函数原型 | **static int SAMPLE\_PRIVATE\_VO\_InitScreen720x1280\_guoxian\_1(void)** |
| 功能描述 | 国显1mipi初始化 |
| 输入参数 | 无 |
| 输出参数 | Int型返回值 |
| 全局变量值的变化 | 无 |
| 返回值 | Int型返回值 |

函数实现描述（自然语言）：

static int SAMPLE\_PRIVATE\_VO\_InitScreen720x1280\_guoxian\_1(void)

{

初始化屏幕；

}

##### SAMPLE\_PRIVATE\_VO\_InitScreen720x1280\_guoxian\_2函数(该类函数依屏幕初始化要求调整)

|  |  |
| --- | --- |
| 函数原型 | **static int SAMPLE\_PRIVATE\_VO\_InitScreen720x1280\_guoxian\_2(void)** |
| 功能描述 | 国显2 mipi初始化 |
| 输入参数 | 无 |
| 输出参数 | Int型返回值 |
| 全局变量值的变化 | 无 |
| 返回值 | Int型返回值 |

函数实现描述（自然语言）：

static int SAMPLE\_PRIVATE\_VO\_InitScreen720x1280\_guoxian\_2(void)

{

初始化屏幕；

}

##### SAMPLE\_PRIVATE\_VO\_InitScreen720x1280\_zhongxian函数(该类函数依屏幕初始化要求调整)

|  |  |
| --- | --- |
| 函数原型 | **static int SAMPLE\_PRIVATE\_VO\_InitScreen720x1280\_**zhongxian**(void)** |
| 功能描述 | 中显1 mipi初始化 |
| 输入参数 | 无 |
| 输出参数 | Int型返回值 |
| 全局变量值的变化 | 无 |
| 返回值 | Int型返回值 |

函数实现描述（自然语言）：

static int SAMPLE\_PRIVATE\_VO\_InitScreen720x1280\_zhongxian(void)

{

初始化屏幕；

}

##### SAMPLE\_PRIVATE\_VO\_InitScreen720x1280\_zhongxian\_2函数(该类函数依屏幕初始化要求调整)

|  |  |
| --- | --- |
| 函数原型 | **static int SAMPLE\_PRIVATE\_VO\_InitScreen720x1280\_**zhongxian\_2**(void)** |
| 功能描述 | 中显2 mipi初始化 |
| 输入参数 | 无 |
| 输出参数 | Int型返回值 |
| 全局变量值的变化 | 无 |
| 返回值 | Int型返回值 |

函数实现描述（自然语言）：

static int SAMPLE\_PRIVATE\_VO\_InitScreen720x1280\_zhongxian\_2(void)

{

初始化屏幕；

}

##### LCD\_Reset函数

|  |  |
| --- | --- |
| 函数原型 | **void LCD\_Reset(void)** |
| 功能描述 | LCD 复位，清除数据 |
| 输入参数 | 无 |
| 输出参数 | 无 |
| 全局变量值的变化 | 无 |
| 返回值 | 无 |

函数实现描述（自然语言）：

void LCD\_Reset(void)

{

LCD复位；

}

##### SAMPLE\_USER\_INIT\_MIPITx函数

|  |  |
| --- | --- |
| 函数原型 | **static int SAMPLE\_USER\_INIT\_MIPITx(cmd\_info\_t \*pcmd\_info)** |
| 功能描述 | mipi数据传输 |
| 输入参数 | 数据地址 |
| 输出参数 | Int型返回值 |
| 全局变量值的变化 | 无 |
| 返回值 | Int型返回值 |

函数实现描述（自然语言）：

static int SAMPLE\_USER\_INIT\_MIPITx(cmd\_info\_t \*pcmd\_info)

{

数据传输；

}

# 系统出错处理

海思系统存在proc目录可以提供mipi\_tx,模块状态实时查看：

