



CV186AH 屏幕对接使用手册

Version: 0.1

Release date: 2023/12

©2022 北京晶视智能科技有限公司
本文件所含信息归北京晶视智能科技有限公司所有。
未经授权，严禁全部或部分复制或披露该等信息。

目录

1	声明	2
2	MIPI DSI	3
2.1	环境准备	3
2.1.1	MIPI DSI 屏幕接口介绍	3
2.1.2	硬件连线确认	4
2.2	配置 MIPI 屏	4
2.2.1	在 u-boot 中配置 MIPI 屏	4
2.2.2	在 kernel 中配置 MIPI 屏	5
2.2.2.1	配置 MIPI Tx 设备属性	5
2.2.2.2	配置屏幕初始化序列	5
2.2.2.3	添加头文件的引用	5
2.2.2.4	配置 MIPI 屏 RESET、POWER、BACKLIGHT 管脚	6
2.2.2.5	编译验证	6
3	LVDS	8
3.1	环境准备	8
3.1.1	LVDS 屏幕接口介绍	8
3.1.2	硬件连线确认	9
3.2	配置 LVDS 屏	9
3.2.1	在 u-boot 中配置 LVDS 屏	9
3.2.2	在 kernel 中配置 LVDS	9
3.2.2.1	配置 LVDS 设备属性	10
3.2.2.2	添加头文件的引用	10
3.2.2.3	配置 LVDS 屏 BACKLIGHT 管脚	10
3.2.2.4	编译验证	10

修订记录

Revision	Date	Description
0.1	2024/01/02	Initial version

1 声明



法律声明

本数据手册包含北京晶视智能科技有限公司（下称“晶视智能”）的保密信息。未经授权，禁止使用或披露本数据手册中包含的信息。如您未经授权披露全部或部分保密信息，导致晶视智能遭受任何损失或损害，您应对因之产生的损失/损害承担责任。

本文件内信息如有更改，恕不另行通知。晶视智能不对使用或依赖本文件所含信息承担任何责任。本数据手册和本文件所含的所有信息均按“原样”提供，无任何明示、暗示、法定或其他形式的保证。晶视智能特别声明未做任何适销性、非侵权性和特定用途适用性的默示保证，亦对本数据手册所使用、包含或提供的任何第三方的软件不提供任何保证；用户同意仅向该第三方寻求与此相关的任何保证索赔。此外，晶视智能亦不对任何其根据用户规格或符合特定标准或公开讨论而制作的可交付成果承担责任。

联系我们

地址 北京市海淀区丰豪东路 9 号院中关村集成电路设计园（ICPARK）1 号楼

深圳市宝安区福海街道展城社区会展湾云岸广场 T10 栋

电话 +86-10-57590723 +86-10-57590724

邮编 100094（北京）518100（深圳）

官方网站 <https://www.sophgo.com/>

技术论坛 <https://developer.sophgo.com/forum/index.html>

2 MIPI DSI

概述

The Display Serial Interface (DSI) 接口是移动行业处理器接口联盟 (Mobile Industry Processor Interface alliance, MIPI 联盟) 定义的一种高速串行接口, 主要用于处理器和显示模块之间的连接。

本章介绍如何在 CVITEK 处理器解决方案上开发调试 MIPI LCD 屏, 以帮助客户有序快速开发 MIPI LCD 业务。

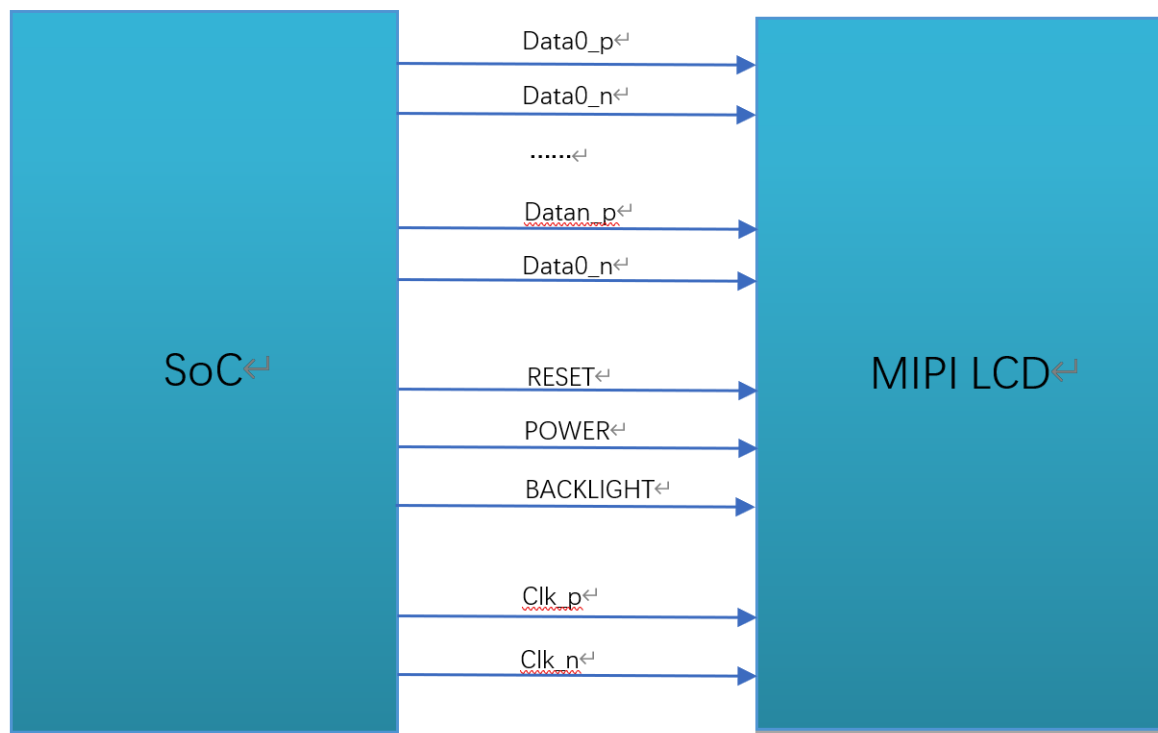
2.1 环境准备

2.1.1 MIPI DSI 屏幕接口介绍

MIPI DSI 屏幕一般有以下几种信号, 如图所示。

- mipi 时钟线 (CLK)
- mipi 数据线 (DATA), 最大为 4Lane (仅可以为 1/2/4Lane)
- 背光控制信号 (BACKLIGHT)
- 复位引脚 (RESET)
- Panel 电源供电 (POWER)

MIPI DSI 接口连线示意图



2.1.2 硬件连线确认

检查硬件连线，确认无异常。具体有些引脚差异，需对照屏幕厂商提供的规格书及电路原理图确认。

2.2 配置 MIPI 屏

根据上节环境准备的内容，在接口和连线上了解了屏幕对接的配置，在这一章节中将对屏幕对接时在软件方面需要进行的配置进行说明。

CVITEK 有两种方案进行 MIPI 屏幕的对接，分别是在 u-boot 及 kernel 中进行屏的初始化，区别在于 u-boot 中进行初始化后，开机可以显示客户的 logo 图片，而带屏的产品基本都会有显示 logo 的需求。实际应用中根据需求二者选其一。

2.2.1 在 u-boot 中配置 MIPI 屏

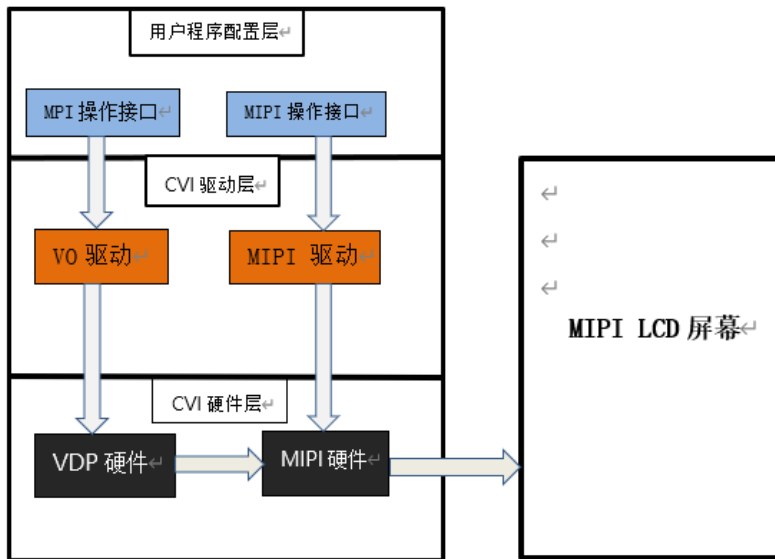
TODO

2.2.2 在 kernel 中配置 MIPI 屏

在 kernel 中配置 MIPI 屏的方法跟在 u-boot 中几乎是一样的，只是实现流程不一样。当无需显示 logo 的时候，可选择此种方式。

另外，也可以先用 kernel 方式调通，再移植到 u-boot，避免频繁烧写 u-boot。

kernel 中对接 MIPI 屏幕基本框图



2.2.2.1 配置 MIPI Tx 设备属性

根据屏的规格书，实现每个屏的配置头文件，并放置在路径 `middleware/component/panel/cv186ah/` 下，客户可以参照其余的头文件模板新增自己的 panel 头文件。

参见 2.2.1.1 节

2.2.2.2 配置屏幕初始化序列

参见 2.2.1.2 节

2.2.2.3 添加头文件的引用

添加对该新增的头文件的引用，在 `middleware/sample/mipi_tx/sample_dsi_panel.h` 中增加对上两节中新增头文件的引用。

示例：

```
#ifndef MIPI_PANEL_HX8394
#include "dsi_hx8394_evb.h"
static struct panel_desc_s panel_desc = {
```

(下页继续)

(续上页)

```
.panel_name = "HX8394-720x1280",
.dev_cfg = &dev_cfg_hx8394_720x1280,
.hs_timing_cfg = &hs_timing_cfg_hx8394_720x1280,
.dsi_init_cmds = dsi_init_cmds_hx8394_720x1280,
.dsi_init_cmds_size = ARRAY_SIZE(dsi_init_cmds_hx8394_720x1280)
};
#endif
```

2.2.2.4 配置 MIPI 屏 RESET、POWER、BACKLIGHT 管脚

方法：

直接在应用层直接控制 GPIO。

示例：

假设 reset: GPIOB5, pwm: GPIOB3, power: GPIOB4, 需要以下操作：

```
1. echo 453 > /sys/class/gpio/export
   echo 451 > /sys/class/gpio/export
   echo 452 > /sys/class/gpio/export
2. echo out > /sys/class/gpio/gpio453/direction
   echo out > /sys/class/gpio/gpio451/direction
   echo out > /sys/class/gpio/gpio452/direction
3. echo 1 > /sys/class/gpio/gpio453/value
   echo 1 > /sys/class/gpio/gpio451/value
   echo 1 > /sys/class/gpio/gpio452/value
   echo 0 > /sys/class/gpio/gpio453/value
   echo 1 > /sys/class/gpio/gpio453/value
```

说明：

为调试方便，背光可先用 GPIO 控制，切记先不要在 u-boot 中配置 pinmux 为 PWM 功能，否则可能无法控制。

后续根据需求，如果需要调节亮度，在 u-boot 中配置 pinmux 功能为 PWM，同时 app 中用 PWM 方式控制。

2.2.2.5 编译验证

执行 build_middleware 编译 middleware，在路径 middleware/sample/mipi_tx/下会生成 sample_dsi 可执行文件。该程序和 u-boot 中 “startvo 0 65536 0” 做的事情是一样的，切换到 LP 模式，设置 MIPI Tx 设备属性并通过 Data Lane0 向屏幕发送初始化序列，然后切回 HS 模式。

将 sample_dsi 拷贝至设备，运行。

说明：

RESET 初始电平设置为 low，将需要 high-low-high 时序变化。

RESET 初始电平设置为 high，将需要 low-high-low 时序变化。

使能 VO 的 test pattern，寄存器如下图。执行 devmem 0x67005094 32 0x0701000a 将会看到 colorbar。

h94	h94	REG_37		rstn	reg_gra_inv		0	0	1	h0		rw	pattern gen gradient invert
h94	h94			rstn	reg_pat_en		1	1	1	h0		rw	pattern gen enable
h94	h94			rstn	reg_auto_en		2	2	1	h0		rw	auto change pattern enable
h94	h94			rstn	reg_dith_en		3	3	1	h1		rw	dither function enable
h94	h94			rstn	reg_snow_en		4	4	1	h0		rw	snow pattern enable
h94	h94			rstn	reg_fix_mc		5	5	1	h0		rw	mde window output fix color enable
h94	h94			rstn	reg_dith_md		10	8	3	h0		rw	dither noise repeat method 0 : no repeat other : (2^reg_dith_md-1)
h94	h94			rstn	reg_pat_prd		23	16	8	h1		rw	auto pattern gen period, unit is frame
h94	h94			rstn	reg_pat_idx		28	24	5	h0		rw	pattern index when reg_auto_en is 0

若 colorbar 未正常显示，请回头检查此前的流程是否设置正确及达到预期。

假如此前流程均未发现异常，建议查看 Driver IC datasheet 或直接咨询屏幕厂商，如何开启屏的 BIST mode，通常是调整初始化序列中的某个寄存器值，会显示 colorbar 等。

假如 BIST mode 不正常，则需要再检查 MIPI Lane 顺序、RESET、POWER、PWM 等是否配置正确，并使用万用表/示波器等确认电路电平状态符合预期，假如均符合预期，则可能是屏幕本身的问题，请咨询屏幕厂商。

假如 BIST 正常，则说明以上配置正确，硬件电路没有异常，这时通常需要调整 sync_info_s 中的各参数。

3 LVDS

概述

Low Voltage Differential Signal(LVDS)，即低电压差分信号。LVDS 接口又称 RS644 总线接口，1994 年由美国国家半导体公司 (NS) 提出的为克服以 TTL 电平方式传输宽带高码率数据时功耗大、EMI 电磁干扰大等缺点而研制的一种视频信号传输模式，是一种电平标准，广泛应用于液晶屏接口。LVDS 屏总体和 MIPI 类似，但是还有一些区别。本章节介绍如何在 CVITEK 处理器解决方案上开发调试 LVDS LCD 屏。

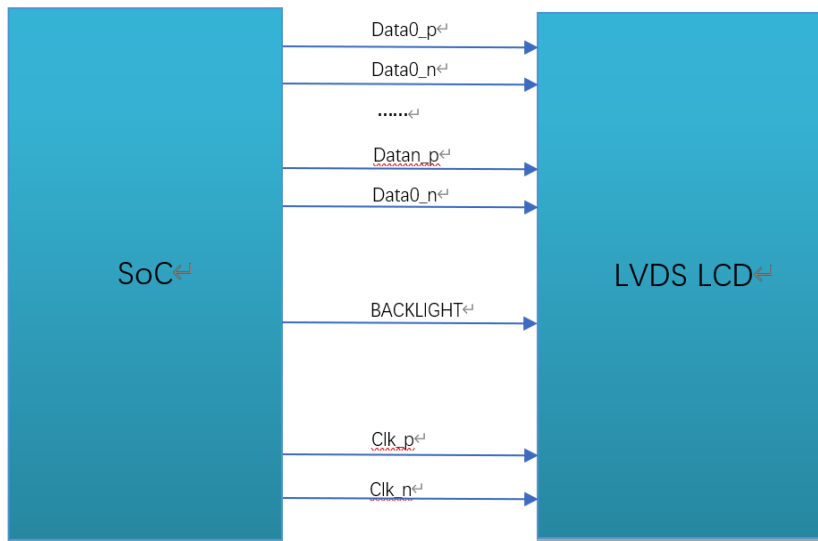
3.1 环境准备

3.1.1 LVDS 屏幕接口介绍

LVDS 屏幕一般有以下几种信号，如图所示。

- LVDS 时钟线 (CLK)
- LVDS 数据线 (DATA) (单路 6bit: 3 lane, 单路 8bit: 4 lane, 单路 10bit: 5 lane, 双路 6bit: 6 lane, 双路 6bit: 8 lane, 双路 6bit: 10 lane, 目前仅支持单路 6bit 和单路 8bit)
- 背光控制信号 (BACKLIGHT)

LVDS 接口连线示意图



3.1.2 硬件连线确认

检查硬件连线，确认无异常。具体有些引脚差异，需对照屏幕厂商提供的规格书及电路原理图确认。

3.2 配置 LVDS 屏

根据上节环境准备的内容，在接口和连线上了解了屏幕对接的配置，在这一节中将对屏幕对接时在软件方面需要进行的配置进行说明。

CVITEK 有两种方案进行 LVDS 屏幕的对接，和 MIPI 屏类似，分别是在 u-boot 及 kernel 中进行屏的初始化。实际应用中根据需求二者选其一。

3.2.1 在 u-boot 中配置 LVDS 屏

TODO

3.2.2 在 kernel 中配置 LVDS

在 kernel 中配置 LVDS 屏的方法跟在 u-boot 中几乎是一样的，只是实现流程不一样。当无需显示 logo 的时候，可选择此种方式。

另外，也可以先用 kernel 方式调通，再移植到 u-boot，避免频繁烧写 u-boot。

3.2.2.1 配置 LVDS 设备属性

根据屏的规格书，实现每个屏的配置头文件，并放置在路径 `middleware/component/panel/cv186ah/` 下，客户可以参照其余的头文件模板新增自己的 panel 头文件。

参见 2.2.1.1 节。

3.2.2.2 添加头文件的引用

添加对该新增的头文件的引用，在 `middleware/component/panel/cv186ah/lvds_panels.h` 中增加对上一节中新增头文件的引用。

示例：

```
#ifndef LVDS_PANEL_EK79202
#include "lvds_ek79202.h"
const VO_LVDS_ATTR_S *pstLvdsAttr = &lvds_ek79202_cfg;
#endif
```

3.2.2.3 配置 LVDS 屏 BACKLIGHT 管脚

在路径 `middleware/component/panel/cv186ah/` 下找到对应的头文件，配置 LVDS 的 gpio 信息，如果没有该管脚或者由 APP 控制，则直接不写或者 `gpio_num` 赋值为 -1 即可。

示例：

```
.backlight_pin = {
    .gpio_num = GPIOE_02,
    .active = GPIO_ACTIVE_HIGH,
},
```

说明：

为调试方便，背光可先用 GPIO 控制，切记先不要在 u-boot 中配置 pinmux 为 PWM 功能，否则可能无法控制。

后续根据需求，如果需要调节亮度，再在 u-boot 中配置 pinmux 功能为 PWM，删除头文件中的此配置或者 `gpio_num` 赋值为 -1，同时 APP 中用 PWM 方式控制。

3.2.2.4 编译验证

LVDS 在未打开 logo 情况下，需运行 APP 才能出图。屏的初始化参见 `middleware/sample/common/sample_common_platform.c`，需修改 `stDefDispRect` 和 `stDefImageSize` 修改为屏实际大小，`stVoConfig.stVoPubAttr.enIntfType` 修改为 `VO_INTF_LCD_24BIT` (`VO_INTF_LCD_18BIT`、`VO_INTF_LCD_30BIT`)，`stVoConfig.stVoPubAttr.enIntfSync` 修改为 `VO_OUTPUT_1280x800_60`，其他部分参见 `middleware/sample/common/sample_common_vo.c` 中 `SAMPLE_COMM_VO_FillIntfAttr` 和 `SAMPLE_COMM_VO_StartDev` 的实现。

运行 APP 中 `CVI_VO_SetPubAttr` 和在 u-boot 中 “startvo 0 2048 0” 做的事情是一样的，初始化 LVDS 屏并让于工屏处作状态。

如未能正常显示，可参见 3.2.1.6.